End-to-END Mission / System Performance Simulation

ECSIM

SYSTEM USER'S MANUAL

(for SW version 1.2)

Code : ECSIM-DMS-TEC-SUM-01-R
Issue : 2.4
Date : 07/05/2008

	Name	Function	Signature
Prepared by	José Julio Ramos Juan R. Acarreta Dave Donovan	Project Engineer	
Reviewed by	Ricardo Moyano Juan R. Acarreta	Review Team	
Approved by	Ricardo Moyano	Project Manager	
Signatures and approvals on original			

DEIMOS Space S.L.
Ronda de Poniente, 19, Edificio Fiteni VI, 2-2ª
28760 Tres Cantos (Madrid), SPAIN
Tel.: +34 91 806 34 50 / Fax: +34 91 806 34 51
E-mail: deimos@deimos-space.com

© DEIMOS Space S.L., 2007

This page intentionally left blank

© DEIMOS Space S.L., 2007

Document Information

Contract Data	
Contract Number:	
Contract Issuer:	

Internal Distribution		
Name	Unit	Copies
Ricardo Moyano	DEIMOS	1
Juan R. Acarreta	DEIMOS	1
Jose Julio Ramos	DEIMOS	1
Internal Confidentiality Level (DMS-COV-POL05)		
Unclassified ☐	Restricted ☑	Confidential ☐

External Distribution		
Name	Organisation	Copies
Raffaella Franco	ESA	1
Dave Donovan	KNMI	1

Archiving	
Word Processor:	MS Word 2003
File Name:	ECSIM-DMS-TEC-SUM01-24-R.doc

© DEIMOS Space S.L., 2007

Document Status Log

Issue	Change description	Date	Approved
1.0	This version is produced for PM2 starting from the draft discussed at PDR	19/04/07	
1.1	Updated sections 4 and 0 to cover RID PM2-RF01. Updated sections 3 and 3.3 to cover RID PM2-RF02. Updated batch mode definition in section 1.3 to cover RID PM2-DL22. Updated simulation definition in section 1.3 to cover RID PM2-DL23. Updated foot note in page **Error! Bookmark not defined.** to include the definition of "vertical dimensions". Major changes in section 4. Screenshots have been updated	15/06/07	
1.2	Included changes all over the document to fulfil PM3 RIDs and actions	03/07/07	
	Fixed typographic error in "Document information – External distribution".	24/09/07	
2.0	Major changes all over the document.	17/10/07	
2.1	Changes to meet ESA's comments on 9th, November 2007: - Section. 3.2.1. Clarified the correspondence between $ECSIM_HOME, the placeholder, and $ECSIM_HOME, the environment variable. - Section 0. Detailed the models that need the scattering libraries. - Section 3.4.1. Added instructions to copy the MySQL connector/J jar file to the system. - Section 3.4.2 completed with instructions to install the scattering libraries and DEM. - Section 3.4.3. Detailed and explained the variables to set before the compilation process. - Section 3.4.4. Changed folders named "auxiliary" to "aux". - Section 3.5. Added a note about the ECSIM executable. - Section 4.16.1. Replaced append (>>) operator by create (>) operator.	15/12/07	

© DEIMOS Space S.L., 2007

2.2	Changes to meet ESA's comments at CDR, on 28th, November 2007: - Updated figures and section 1 to match software version 1.1 - Updates tables of applicable and reference documents to match latest versions. - Deleted former section 3.5.1 describing the setup of environment variables. Included instructions for setting those variables through the installation script. - Added an introduction to section 4.17. - Included a new section after number 4.17 describing the error messages generated by the framework and models. - Updated an extended description of the scattering libraries. - Removed section "Scenes. Generating XML". - Included instructions to import scenes from third-party applications. - Fixed section 4.9.2 Simulation creation to show the correct way to access the functionality. - Table in section 4.9.2.1. "Simulation ID" in first row, first column changed to "Identifier". - Updated section 4.11.2.2 to correctly name the "add" and "remove" buttons. - Updated section 5 to show latest changes in the repository. - Included more information about locating file instances (when editing a session) by absolute paths or $ECSIM_HOME relative paths. - Added a new session to describe nominal (included in the distribution) scenes. - Added instructions to edit numerical sequences when iterating parameters in a session creation. - Updated figures and text to show the changes in the iterative session feature: input and output files cannot be iterated anymore. - Simplified surface types. - Added detailed information about using environment variables when scheduling product tools in a simulation.	08/01/08		

© DEIMOS Space S.L., 2007

2.3	Changes to meet ESA's comments at AR, on 6th, March 2008: - Following AR-RF-07, updated titles of section 5.6 and 5.7 and contents of section 5.6 to clarify that the two scenes are just two examples given for testing purposes in the final delivery. - AR-DL-11. Updated figures to match current SW. - AR-DL-12. Updated section 3.4.4 to show the current and actual folder structure. - AR-SP-13. Section 3.3.4. Add "or superior" in the Intel Fortran compiler version identifier. - AR-SP-14. Updated table in section 3.3.3 to show the exact version of the Connector/J and added instructions in section 3.4.1 in case users want to use a different version. - AR-SP-15. Changed section 3.4.2 to separate the scattering libraries and the DEM. Added clarifications about the minimal set and instructions to build the complete suite. - AR-SP-16. Corrected section 3.4.2.2. The DEM is not provided as a tar ball. - AR-SP-17. Substituted "General data" with "General properties" in section 4.10.2.2. - AR-SP-18. Changed "pressing the 'accept' button" to "pressing the 'Ok' button" at the end of section 4.10.2.9 - AR-SP-19. Deleted section 4.11.3 Open scene because that functionality is no longer in the system. - AR-SP-20. Updated section 4.11.6. Deleted "TBD" - AR-ME-34. Included in section 3.3.3 and 3.3.4 a note to include the executables of some pre-requisites into the $PATH system variable. - AR-ME-35. Removed from section 3.4.3 the mention to ECSIM_HOME. Clarified in section 4.5.1 that model executions get passed the environment variables. - AR-ME-36. Included PGPLOT_FFLAGS examples. - AR-ME-38. Updated section 3.4.3. CFI installation is mandatory, so "should" is replaced by "must". - AR-ME-39. Removed comment on figure in section 4.5.2.1 saying "no parameter passing at this stage". - AR-ME-41. Added detailed information in 4.10.2.8 and 4.15.1 explaining the variable passing technique.	31/03/08		

© DEIMOS Space S.L., 2007

2.4	Changes to meet ESA's comments at AR closeout: ❑ Updated section 3.3.3 and section 3.3.4 to • Fix the Fortran Compiler version • Remind to set the root user name and password after the MySQL server installation • Stick to the SUN Java RE. ❑ Updated figure 3.1 to show a more simplistic block diagram		

© DEIMOS Space S.L., 2007

Table of Contents

1. Introduction and purpose 15
1.1. Scope 15
1.2. Acronyms and Abbreviations 15
1.3. Definitions 16
2. Related documents 20
2.1. Applicable Documents 20
2.2. Reference Documents 20
2.3. Standards 20
3. Getting started 21
3.1. Introduction 21
3.2. Conventions used in this Manual 24
3.2.1. $ECSIM_HOME 24
3.2.2. Data types 24
3.3. Initial Requirements 24
3.3.1. Hardware requirements 24
3.3.2. Operating system requirements 24
3.3.3. Framework pre-requisites 25
3.3.4. Models and product tools minimum pre-requisites 25
3.3.5. Models and product tools optional components 26
3.4. How to Install the Application 28
3.4.1. Framework installation 28
3.4.2. Scattering libraries and DEM installation 29
3.4.2.1. Scattering libraries 29
3.4.2.2. DEM 30
3.4.3. Model compilation 30
3.4.4. Folder structure 31
3.4.5. Licensing scheme 32
3.5. How to Start the Application 33
3.5.1. Environment variables 33
4. ECSIM – Reference manual 34
4.1. Main window 34
4.2. Generic Functions, dialogues and displays 36

© DEIMOS Space S.L., 2007

4.3. Frame management 37
4.4. Side bar 38
4.5. System 38
4.5.1. Show configuration 40
4.5.2. Tools 42
4.5.2.1. New tool 42
4.5.2.2. Edit tool 43
4.5.2.3. Delete tool 43
4.5.3. About ECSIM 43
4.5.4. Exit the system 43
4.6. Repository 44
4.7. Descriptors 45
4.7.1. Descriptor list 45
4.7.2. Descriptor creation 46
4.7.3. Descriptor edition 47
4.7.4. Descriptor deletion 47
4.8. Models 48
4.8.1. Model developers guideline 49
4.8.2. Model list 49
4.8.3. Adding models 50
4.8.3.1. General data 50
4.8.3.2. Configuration 52
4.8.3.3. IO descriptors 53
4.8.4. Model upgrade - New version 54
4.8.5. Model edition 55
4.8.6. Model deletion 55
4.9. Simulations 55
4.9.1. Simulation list 56
4.9.2. Simulation creation 56
4.9.2.1. General properties 57
4.9.2.2. Models schema 58
4.9.3. Simulation creation example: creating a forward simulation of the active instruments 59
4.9.3.1. Step 1: Entering the model list tab 59
4.9.3.2. Step 2: Selecting the first model of the simulation 60
4.9.3.3. Step 3: Selecting the remaining models of the simulation 61

© DEIMOS Space S.L., 2007

4.9.3.4. Step 4: Finishing the simulation creation 63
4.9.4. Simulation edition 64
4.9.5. Simulation deletion 64
4.9.6. Simulation execution 65
4.10. Sessions 65
4.10.1. Session list 66
4.10.2. Session creation 67
4.10.2.1. Import a session folder 67
4.10.2.2. General properties 67
4.10.2.3. Altering the simulation set 68
4.10.2.4. Provision of input data 69
4.10.2.5. Provision of configuration files 70
4.10.2.6. Parameters configuration 70
4.10.2.7. Specification of output files 71
4.10.2.8. Specification of final product tools 72
4.10.2.9. Iterative sessions – iterating input/output files and parameter values 73
4.10.3. Session edition 76
4.10.4. Session deletion 76
4.10.5. Session execution – run 76
4.10.6. Session script generation 79
4.11. Scenes 79
4.11.1. List scenes 79
4.11.2. New scene 80
4.11.2.1. General data 80
4.11.2.2. Scene dimensions 81
4.11.2.3. Atmosphere composition 82
4.11.2.4. Scene surfaces 84
4.11.2.5. Clouds and aerosols 86
4.11.2.5.1. Common region attributes 87
4.11.2.5.2. Non fractal region attributes 88
4.11.2.5.3. Fractal region attributes 89
4.11.3. Delete scene 91
4.11.4. Compare scenes 92
4.11.5. Import scene 92
4.11.6. Merge scene 92

© DEIMOS Space S.L., 2007

4.12. Executions 93
4.13. Results 93
4.13.1. Result view 94
4.13.2. Result re-run 96
4.13.3. Report generation 96
4.13.4. Delete result 97
4.14. Logs 97
4.14.1. Log messages list 97
4.15. File system 98
4.15.1. Tool execution 99
4.16. Persistent storage - Database and file system 100
4.16.1. Database maintenance 101
4.17. Table of acceleration keys 101
4.18. Error messages 103
5. ECSIM – operations manual 109
5.1. Forward branch simulation 109
5.1.1. Operation objectives 109
5.1.2. Pre-conditions 109
5.1.3. Steps 110
5.2. Single Instrument retrieval 110
5.2.1. Operation objectives 110
5.2.2. Pre-conditions 110
5.2.3. Steps 111
5.3. Synergistic Instrument Retrieval 111
5.3.1. Operation objectives 111
5.3.2. Pre-conditions 111
5.3.3. Steps 112
5.4. Experiment with Multiple runs 113
5.4.1. Operation objectives 113
5.4.2. Pre-conditions 113
5.4.3. Steps 114
5.5. E2E simulation 114
5.5.1. Operation objectives 114
5.5.2. Pre-conditions 116
5.5.3. Steps 117

© DEIMOS Space S.L., 2007

5.6. List of available scenes.....118
5.7. List of available tools.....118

List of Figures

Figure 3-1: Full ECSIM schema.....23
Figure 4-1: Main window appearance.....34
Figure 4-2: Main window appearance showing internal frames and scroll bars.....35
Figure 4-3: Detail of main menu bar.....36
Figure 4-4: Detail of a menu, showing menu items.....36
Figure 4-5: Detail of a contextual menu.....36
Figure 4-6: File chooser dialog.....37
Figure 4-7: Dialog example.....37
Figure 4-8: Frame management menu.....38
Figure 4-9: Internal frame header.....38
Figure 4-10: System menu.....39
Figure 4-11: Side bar.....39
Figure 4-12: System configuration.....40
Figure 4-13: Tools list view.....42
Figure 4-14: Tool. Creation.....42
Figure 4-15: ECSIM logical flow.....44
Figure 4-16: Repository view.....44
Figure 4-17: Descriptor. Side bar.....45
Figure 4-18: Descriptor list view.....46
Figure 4-19: Descriptors. New descriptor.....47
Figure 4-20: Repository menu.....48
Figure 4-21: Model. Pop-up menu.....48
Figure 4-22: Model list view.....49
Figure 4-23: Model. General properties.....52
Figure 4-24: Model. Configuration.....53
Figure 4-25: Model. Input/Output specification.....54
Figure 4-26: Simulations menu.....55
Figure 4-27: Simulation. Pop-up menu.....55
Figure 4-28: Simulations list view.....56

© DEIMOS Space S.L., 2007

Figure 4-29: Simulation. Creation 57
Figure 4-30: Defining a simulation - initial model window 60
Figure 4-31: Defining a simulation - selection of the first model 61
Figure 4-32: Defining a simulation – presentation of consistent models 62
Figure 4-33: Defining a simulation – selecting models 63
Figure 4-34: Defining a simulation – final window 64
Figure 4-35: Simulation. Execution 65
Figure 4-36: Session menu 66
Figure 4-37: Session pop-up menu 66
Figure 4-38: Session list view 67
Figure 4-39: Session import 67
Figure 4-40: Session creation – General properties 68
Figure 4-41: Session – Simulations specification 69
Figure 4-42: Session – Inputs definition 69
Figure 4-43: Session – Configuration files definition 70
Figure 4-44: Session – Parameters definition 71
Figure 4-45: Session – Output definition 72
Figure 4-46: Session – Product tools specification 72
Figure 4-47: Session creation – Iterating parameters 74
Figure 4-48: Session - Creation - Editing numeric sequences 74
Figure 4-49: Session creation – Pre-viewing the parameters iteration 75
Figure 4-50: Session execution – Redundancy 77
Figure 4-51: Session – Execution progress 78
Figure 4-52: Scenes main menu 79
Figure 4-53: Scenes tab in the side bar 79
Figure 4-54: Scenes pop-up menu 79
Figure 4-55: Scenes list view 80
Figure 4-56: Scene. General data 81
Figure 4-57: Scene. Vertical resolution 81
Figure 4-58: Scene. Dimensions and resolutions 82
Figure 4-59: Scene. Atmosphere composition 83
Figure 4-60: Scene. Extra gas 83
Figure 4-61: Scene – Surfaces 84
Figure 4-62: Scene. Surface region 85
Figure 4-63: Scene. Clouds and aerosols 86

© DEIMOS Space S.L., 2007

Figure 4-64: Scene. Non fractal scattering region 87
Figure 4-65: Scene. Fractal scattering region 91
Figure 4-66: Scene Comparison 92
Figure 4-67: Scene. Merge 93
Figure 4-68: Side bar. Executions 94
Figure 4-69: Results menu 94
Figure 4-70: Results pop-up menu 94
Figure 4-71: Execution results 95
Figure 4-72: Result. Re-run 96
Figure 4-73: Execution report 96
Figure 4-74: Logs menu 97
Figure 4-75: Logs list view 98
Figure 4-76: File system view 99
Figure 4-77: IO file pop-up menu 100
Figure 5-1: End-to-end simulation schema 115

© DEIMOS Space S.L., 2007

1. INTRODUCTION AND PURPOSE

This document has been produced by DEIMOS within the frame of the ECSIM project and represents the Software User Manual for the ECSIM platform.

It provides not only a clear description on the functionality of the application itself, but is also an operational guide to some of the procedures within an end-to-end simulation of the EarthCARE mission.

This document is applicable to version1.2 of the ECSIM application.

Chapter 1, this present chapter, talks about the document, giving a description and settling the basis to understand it.

Chapter 2 links this document with information from other sources.

Chapter 3 details the procedures for setting the ECSIM system up.

Chapter 4 describes one by one all the different functionalities of the ECSIM system.

Chapter 5 presents some examples of the simulations that can be achieved.

Reading the chapters in this order will help users to fully understand the use of the ECSIM system. Chapter 5 can be skipped by experienced users.

1.1. Scope

The applicability of this document begins once ECSIM starts to be operable. However, it is aimed to support two different kinds of users:

- Users that want to take advantage of the application.
- Developers that want to implement new models intended to be used within ECSIM.

1.2. Acronyms and Abbreviations

Acronym	Description
AD	Applicable Document
API	Application Programming Interface
BB	Broad-Band
BBR	Broad-Band Radiometer
CFI	Customer Furnished Item
COTS	Commercial Off-The-Shelf
DBMS	Database Management System
DMS	DEIMOS Space
E2E	End to end simulation
GUI	Graphical User Interface

© DEIMOS Space S.L., 2007

Acronym	Description
HMI	Human-Machine Interface
HW	Hardware
I/F	Interface
I/O	Input/Output
ICD	Interface Control Document
IT	Integration Test
IWC	Ice water content
KNMI	Royal Netherlands Meteorological Institute
LES	Large Eddy Simulation.
LW	Long-Wave (4.0 to 400.0 μm)
MC	Monte-Carlo technique for Radiative transfer calculations
MMI	Man-Machine Interface
MS	Multiple Scattering
MSC	Meteorological Service of Canada
RD	Reference Document
RDBMS	Relational Data Base Management System
RT	Radiative Transfer
SUM	System User Manual
SW	Short-wave (0.2 to 4.0 μm)
TBC	To Be Confirmed
TBD	To Be Defined / Decided
TN	Technical Note
UML	Unified Modelling Language

1.3. Definitions

Definition	Meaning
Auxiliary data	These data refer to geophysical data produced by third parties and required as input for every kind of model (e.g., ground albedo). For the aim of EarthCARE, the auxiliary data are not dynamic, so they are constant through any given simulation. As a special case, auxiliary data also encompasses scattering libraries during the generation of UFF files.

© DEIMOS Space S.L., 2007

Definition	Meaning
Batch mode	It is the capability of the simulator to perform consecutive runs without a continuous interaction with the user. Batch mode checks the agreement or not between the output of a given module and the input by the next one in the sequence of the simulation. Several modes of executions can be performed: ❑ Iteratively, executing one or more simulations ❑ Iteratively, executing the same simulation several times depending on the parameters configuration ❑ Same as above but by executing a batch script.
Closure	Internal check of consistency following different branches available in the simulator. In principle, it is aimed only for complete simulations (UFF → 3D reconstruction). In such a situation: ❑ High level closure is made comparing TOA fluxes ❑ Low level closure is made comparing UFF and 3D reconstructed scene.
Configuration File	A small XML file that contains all the parameters necessary to execute a model. A configuration file instance must comply with the corresponding XML schema defined at model creation time.
Framework	Software infrastructure designed to support and control the simulation definition and execution. It includes the GUI, domain and database capabilities that enable to perform all the functionality of the ECSIM simulator. Models are not considered part of the framework.
Forward models	Models that transform the scene (or UFF file) into output suitable for the instrument models. The forward models are based on radiative transfer techniques adapted for active and passive instruments
Instrument models	Four types: Radar (CPR), Lidar (ATLID), Imager (MSI) and Radiometer (BBR). These models have as output L0 and L1 products (including L1b and L1c)
Model	Executable entity that can take part in a simulation. A model can be understood, broadly speaking, also as an "algorithm". Basically, it contains the recipe to produce products function of inputs. A model contains also several rules to define the input, output and associated formats. Furthermore, its behaviour is controlled by one configuration file. Overall, the architecture of a model consists of: ❑ The source code and its binary compiled counterpart ❑ A configuration file with its parameters ❑ An input file that characterizes its inputs ❑ An output file that characterizes its outputs In addition, it must belong to one of these classes: scene, forward, retrieval, platform, instrument
Parameter	A constant whose value characterizes a given particularity of a model. Parameters are user-configurable, they are fixed before launching a model and, for practical reasons, not all of them shall be accessible from the HMI.

© DEIMOS Space S.L., 2007

Definition	Meaning
Platform models	Models that navigate the instruments along the orbit. These models are responsible for the proper location (in space) and pointing (towards the earth) of the instruments. Note that actually there is no management of the physical dimensions of a "platform" or "spacecraft". All the instruments will be considered located in an orbital point.
Product (or processing) level	An identifier representing the amount of processing of the instrumental output: ❑ L0: raw data from the instruments. Not used within ECSIM version 1.2. ❑ L1a: Level 0 contents with corresponding radiometric, spectral and geometric (i.e. Earth location) correction and calibration computed and appended, but not applied. Not used within ECSIM version 1.2. ❑ L1b: Fully geo-localised products, as well as error estimates and reliability data for each data set. The data shall be calibrated to engineering units using the best possible characterisation data available. ❑ L1c: It is applicable only to the MSI instrument. It consists of L1 MSI data re-sampled to a specified spatial grid. It shall be possible to use different methods for image re-sampling. Among them there shall be available at least • Bi-cubic convolution interpolation • Nearest neighbour. ❑ L2a: Single instrument science products. ❑ L2b: Science products using more than one instrument in synergy In some circumstances L1(a, b, …) and L2(a, b, …) are also written as L1 and L2 through the main text.
Retrieval models	Models that transform the calibrated output of the instruments into geophysical products (represented also with the keyword L2). They are sometimes known as backward models
Scene	Ensemble of atmospheric data that defines the initial conditions of a simulation. A scene is composed by four basic ingredients: ❑ Ground: It contains the ground type and associated properties (e.g. if selected 'ocean' the user does not need to input its albedo). In addition, the ground elevation is considered. Other properties could be ice / snow masks. ❑ Atmosphere: it includes the pressure-temperature profile and the amount and distribution of several absorbing gases. Aimed for Radar / ocean issues, it also includes wind speed ❑ Aerosols: small particles suspended in the atmosphere that are not either liquid water or solid ice, and that are distributed homogeneously in a given 3D region. ❑ Clouds: collection of condensed liquid water or ice crystals with a regular or fractal spatial distribution. Clouds and aerosols can overlap and intersect. Note that scenes are understood generically as binary files with UFF format. This is why, within the ECSIM framework, the words "scene" and "UFF" are used sometimes as synonyms.

© DEIMOS Space S.L., 2007

Definition	Meaning
Session	A session is defined as an execution of a simulation, an ordered set of simulations or an iterative execution of simulation(s) with different parameter values. There are no restrictions on how to concatenate these simulations, they do not have to be compatible between them but, if necessary, the final output files of a simulation can be used by the following simulation.
Simulation	A simulation is understood as a list of models (or even a model alone) that is run sequentially and produces observable results. A complete simulation would encompass a model per each of the model types defined within ECSIM: Scene → Platform → Forward → Instruments → Retrieval Simulations can be more simple, e.g., Instruments→ Retrieval Simulations can be started and ended at any stage, even using only one model. Interestingly, simulations can be run in groups. Equivalently, simulations can be linked together. This procedure is actually called session, to better identify that we are dealing with groups of simulations. The output of the simulations (and the input also) will be analysed, if required, with product tools. Those will contain editors, viewers, and interfaces to the plotting libraries. It is noted that those third party options are not fixed in general. Exceptions are the plotting libraries, based on PGPLOT, which will provide ECSIM with basic plotting capabilities (to browse and plot an UFF file, for instance). For this purpose, ECSIM will contain configurable variables, accessible through the MMI, with editors or viewers. However, it is assumed that the third-party editors and viewers will be installed (downloaded) by the user.
Synergy	Synergy is essentially a synonym for the L2b products. Basically, it consists of combining the performances of each instrument to obtain state-of-the-art 2D or 3D retrievals).
Tool	A tool is an external executable file that performs a given action to a certain group of files. Used into the ECSIM platform and associated to a certain file extension these tools can be called to perform off-line operations to products involved in simulations.
UFF File	Binary file used by the current simulator containing the scene description. Refer to [RD SW] for further details.
Version	New instance of an existing model where either the source (and binary) code has been updated or the configuration file schema has been changed. Instead of updating its version, a model is considered as new if either its input and output schema is updated. The justification for this approach considers that it is only with I/O that a model is really embedded in the ECSIM architecture.

© DEIMOS Space S.L., 2007

2. RELATED DOCUMENTS

2.1. Applicable Documents

Reference	Code	Title	Issue
[AD RB]	TEC-SW/ECARE-ETE/RB	End-to-End Mission/System Performance Simulator for EarthCARE – Requirements Baseline	1.2
[AD SOW]	ECARE-ETE/SOW	Statement of Work End-To-End Mission/System Performance Simulator for EarthCARE	1.0
[ESIM-V1-REPORT]	Esa: Contract No. 15346/01/NL/MM	The EarthCARE Simulator: Users Guide and Final Report	Issue 1, 13/12/2004
[AD SRD]	ECSIM-DMS-SRD-001	ECSIM: System requirements document	1.5
[AD ADD]	ECSIM-DMS-ADD-001	ECSIM High-Level Architecture Document	1.3

2.2. Reference Documents

Reference	Code	Title	Issue
[RD UML]	ISBN 0-201-57168-4	The Unified Modelling Language User Guide, Grady Booch, James Rumbaugh, Ivar Jacobson.	-
[RD DDVP]	ECSIM-DMS-DDVP-001	ECSIM Design, Development and Validation Plan	2.1
[RD TN]	ECSIM-DMS-TN-001	Technical Note related to the proposed ECSIM Architecture	1B
[RD SW]	ESA contract 15346/01/NL/MM	EarthCare Simulator: Users Guide and Final Report	December 13, 2004
[RD ICD]	ECSIM-KNMI-ICD-001	ECSIM Interface Control Document	1.4
[RD STS]	ECSIM-DMS-STS-001	ECSIM System Tests Specification	1.5

2.3. Standards

Reference	Code	Title	Issue
[ECSS-40-1B]	ECSS-E-40 Part 1B	Software – Part 1: Principles and requirements	1B
[ECSS-40-2B]	ECSS-E-40 Part 2B	Software — Part 2: Document Requirements Definitions (DRDs)	2B
[STD SOW]	ECARE-ETE/SOW	Annex 1 of [AD SOW] – ECSS-E-40 Tailoring	1.0

© DEIMOS Space S.L., 2007

3. GETTING STARTED

3.1. Introduction

The objective of the EarthCARE simulator (ECSIM) is to provide an End-to-End simulation capability for EarthCARE to assess whether the scientific goals and mission requirements can be met.

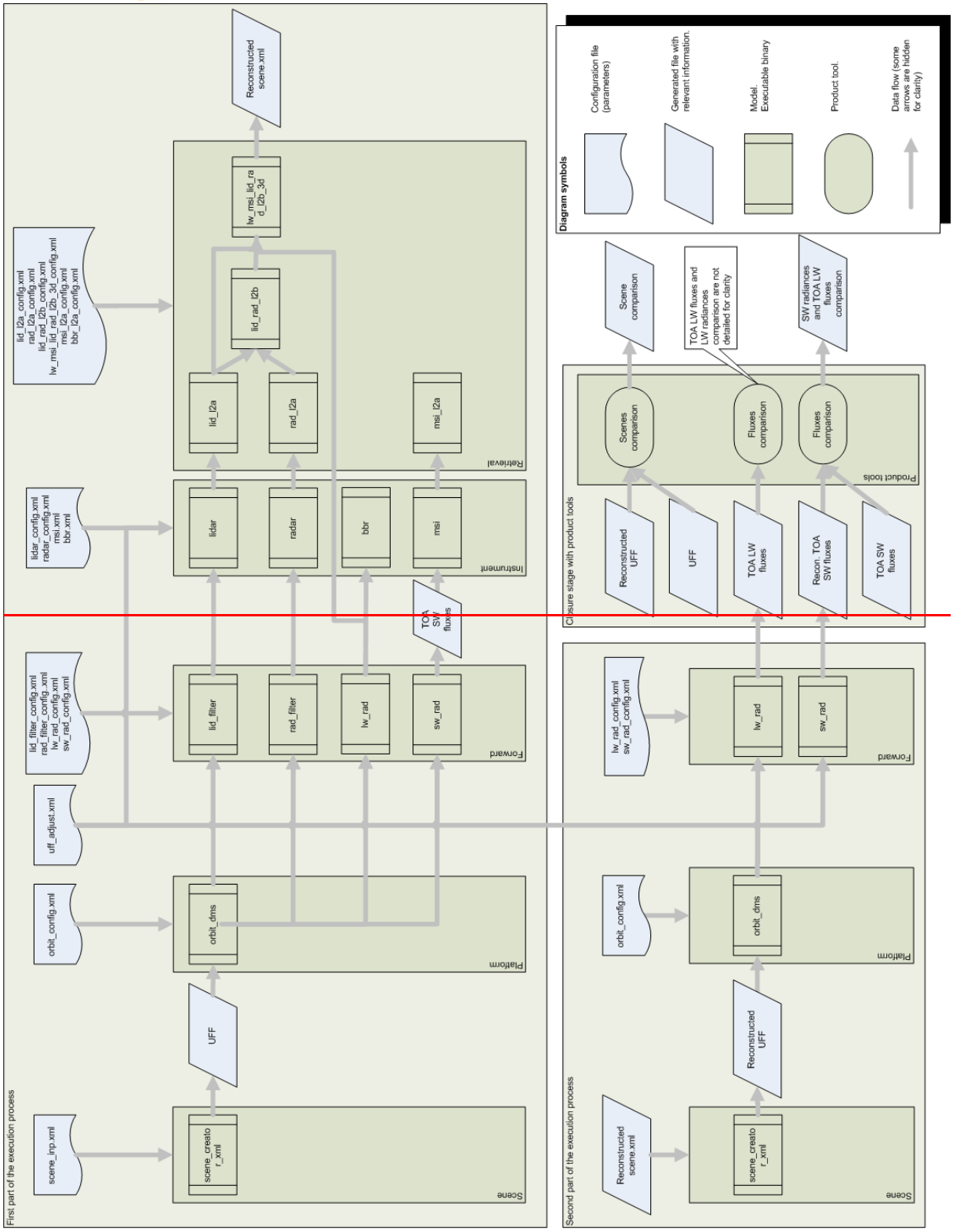
ECSIM is based on an existing E2E simulator developed by KNMI, which is a software system that runs a fixed set of models with no possibility of including new models without a difficult and challenging adaptation process. ECSIM is therefore an upgrade in order to:

- Improve the current engineering and scientific models in scope and quality to provide the necessary functionality for the proper performance assessment for the mission, in nominal conditions and in presence of failures.
- Test different L2-based algorithms that rely on active and passive instruments.
- Consolidate the software engineering approach and architecture, in order to:
  - Allow the inclusion of future instrument/platform/system and data retrieval models by providing clearly engineered model interfaces.
  - Improve the operability and stability of the simulator.
  - Ensure batch processing capability.
  - Ensure computational efficiency.
  - Address validation aspects in collaboration with the scientific user and model developer community.
  - Establish proper configuration control mechanisms.

Such an ambitious scenario can be summarized in the Figure 3-1. The first thing to notice is that any simulation (roughly, any procedure that entails the connection of algorithms) is decomposed into basic entities (boxes called "models" in the figure) with clear I/O. This is completely new since the previous simulator did not have a clear definition of "a model". We can then connect the different models to perform simulations or to perform internal tests (for example through the reconstruction of the input scenario at the end of the simulations). It is also possible to focus on only one model, calling it several times with slightly different conditions (e.g., different configuration parameters) to achieve an efficient sensitivity analysis (or error assessment).

Figure 3-1 indicates all the models and basic closure options available at the time of delivering ECSIM. Therefore, this SUM will focus only on those models and tools. Notwithstanding, ECSIM has been designed to ease the addition of new models. This is also indicated in the current document. Returning again to the figure, it should be added that, as any user would appreciate, it is possible to access directly to any of the processing stages that the simulator has. That is, someone interested only on checking L2 algorithms does not have necessarily to execute, for example, forward models prior to the instruments, unless considered necessary for the generation of suitable and consistent L1 test data. As a result, any user can avoid those areas where he/she has no interest.

© DEIMOS Space S.L., 2007



© DEIMOS Space S.L., 2007

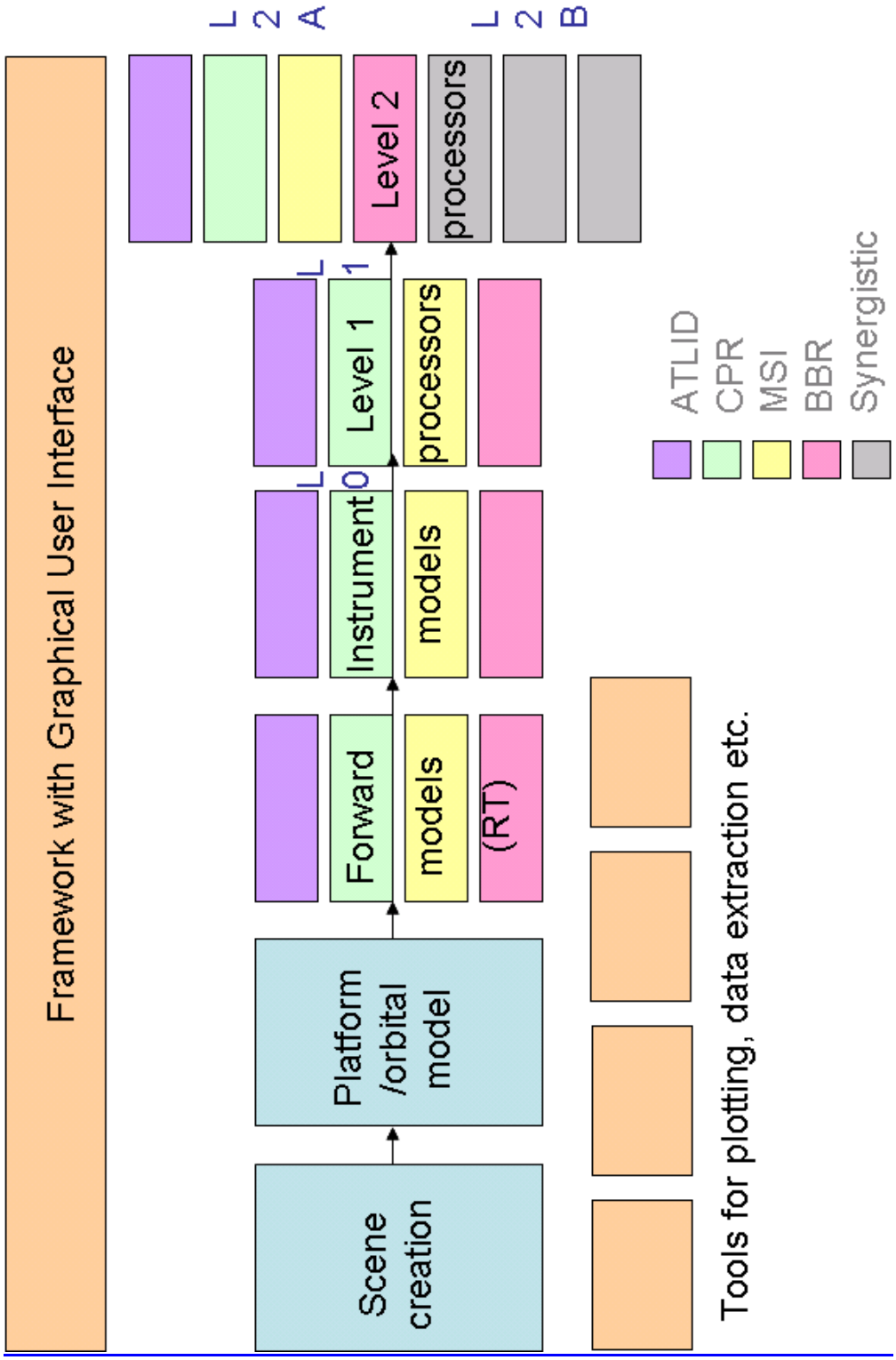


Figure 3-1: Full ECSIM schema

© DEIMOS Space S.L., 2007

3.2. Conventions used in this Manual

This chapter lists all the conventions used throughout this Software User Manual.

3.2.1. $ECSIM_HOME

All through the contents of this User Manual, a "variable" called $ECSIM_HOME is exhaustively used as a placeholder. This variable value points to the root folder that contains the ECSIM installation. Typically, this folder could be similar to this:

```
/usr/home/user_name/ECSIM
```

This variable matches with one "environment variable" defined in the SW. You can find more information in section 4.5.1.

3.2.2. Data types

Every requested or given piece of data in ECSIM is formatted in one of the following data types:

- *Short string*. A string of alphanumeric characters with a size not greater than 25.
- *Medium string*. A string of alphanumeric characters with a size not greater than 75.
- *Long string*. A string of alphanumeric characters with a size not greater than 255.
- *Unsigned integer*. Integer number (no decimal part) between 0 and 2^{31-1}
- *Signed integer*. Integer number (no decimal part) between -2^{31} and 2^{31-1}
- *Float*. Decimal number between 2^{-149} and $(2-2^{-23})\ 2^{127}$
- *Boolean*. TRUE of FALSE.
- *File*. The absolute (or $ECSIM_HOME-relative) path and name of a file into the file system.
- *Folder*. The absolute (or $ECSIM_HOME-relative) path and name of a folder into the file system.

3.3. Initial Requirements

The ECSIM system is prepared to run in a hardware and software platform with the following requirements. These must be fulfilled before installing the distribution.

3.3.1. Hardware requirements

Hardware must at least fulfil the following requirements:

- 32-bit 2GHz dual-core processor
- 2 GB of RAM memory installed
- 12 GB of free space to install the complete distribution (including DEM, scattering libraries and rest of auxiliary files).

3.3.2. Operating system requirements

- Linux. Kernel version 2.6

© DEIMOS Space S.L., 2007

3.3.3. Framework pre-requisites

Pre-requisite	Purpose	Licensing	Distribution site
Sun Java(TM) 2 Runtime Environment, Standard Edition 1.5 or superior	ECSIM runs within this execution environment.	GNU GPL / Java Community Process	http://www.java.com/en/download/
MySQL client and server 5 or superior	ECSIM stores information in this relational database	GPL or Proprietary License	http://dev.mysql.com/downloads/mysql/5.0.html
MySQL connector/J 5.0.4	ECSIM uses this library to connect to the MySQL database server.	GPL or Proprietary License	http://dev.mysql.com/get/Downloads/Connector-J/mysql-connector-java-5.0.4.tar.gz/from/pick

Don't use a Java Runtime Environment from another provider different than Sun.

Remember to set root user name and password after installing the MySQL server.

Note that after the installation of the Sun Java 2 RE and the MySQL client and server, your $PATH system variable must contain the folder location of their main executables.

3.3.4. Models and product tools minimum pre-requisites

Pre-requisite	Purpose	Licensing	Distribution site
Intel Fortran compiler for Linux 9~~ or superior~~	This is the recommended Fortran compiler for the model sources. Only version 9.	Several options. There is a free edition for the community.	http://support.intel.com/support/performancetools/fortran/linux/
GNU C/C++ compiler for Linux v.4.0 or superior	This is the recommended C/C++ compiler for the model sources.	GNU General Public License, GNU Lesser General Public License	http://gcc.gnu.org/
PGPLOT 5	PGPLOT is a device-independent graphics subroutine library. This is used by some product tools to visualize graphics.	Freely available for non-commercial use	http://www.astro.caltech.edu/~tjp/pgplot/ (See note below)

© DEIMOS Space S.L., 2007

Pre-requisite	Purpose	Licensing	Distribution site
NETCDF 3.6	NetCDF (Network Common Data Form) is a set of software libraries and machine-independent data formats that support the creation, access, and sharing of array-oriented scientific data. Many product files used this format	Freely available	http://www.unidata.ucar.edu/software/netcdf/ (See note below)
ESA- Earth Explorer CFI software	Collection of multiplatform precompiled C libraries for timing, coordinate conversions, orbit propagation, satellite pointing calculations, and target visibility calculations, specifically parametrized and configured for EO satellites.	[ESA]	http://eop-cfi.esa.int/CFI/ee_cfi_software.html

Note that the PGPLOT source code should be compiled on the target machine to obtain the 'libpgplot.a' library. It needs as pre-requisite the "png_devel" and "xorg_x11_devel" packages. The following drivers should be installed: GIF, PS, CPS, VCPS, PS and NULL.

Note that also the NETCDF source code should be recompiled with the Fortran compiler used to compile ECSIM.

After the installation of the Intel Fortran and the GNU C/C++ compilers, your $PATH system variable must contain the folder location of ECSIM main executables.

3.3.5. Models and product tools optional components

If the user has already fulfilled all the previous pre-requisites, he/she is already able to install and execute the ECSIM platform, scientific models and tools. However, some of the capabilities are disabled until the following components are present into the system.

Component	Purpose	Licensing	Distribution site
FFTW 2.1	FFTW, for "Fastest Fourier Transform in the West," is a software library for computing discrete Fourier transforms. This is required by CloudGen and used by the "scene_creator" model to generate fractal clouds.	GPL and commercial	http://www.fftw.org/

© DEIMOS Space S.L., 2007

Component	Purpose	Licensing	Distribution site
CloudGen 1.3	Cloudgen is a program for generating three-dimensional stochastic clouds (specifically cirrus) with realistic horizontal and vertical structure. This is used by the "scene_creator" model to generate fractal clouds. Refer to section 4.11.2.5.1 for more information.	GPL	http://www.met.reading.ac.uk/clouds/cloudgen/
Scattering libraries	Scattering phase function data, extinction, and absorption data covering wavelengths from 200 nm to 4000 microns as well as radar reflectivity at 95 GHz. This is needed by all the models but the "Orbit_dms".	BSD	[Distributed with ECSIM]
DEM	Digital Elevation Model. Used by the orbital model.	[ESA]	http://earth.esa.int/services/amorgos/download/getasse/
ncBrowse	Generic netCDF file viewer that includes Java graphics, animations and 3D visualizations for a wide range of netCDF file conventions.	[See official site]	http://www.epic.noaa.gov/java/ncBrowse/
HDFView	Visual tool for browsing and editing NCSA HDF4 and HDF5 files.	[See official site]	http://hdf.ncsa.uiuc.edu/hdf-java-html/hdfview/
Postscript files viewer[1]	Views the plots generated as result of many models.	-	-
Text files viewer[2]	Views the text content of many files (including XML)	-	-
Diff – text files differencer[3]	Highlights the differences between two files.	GPL	-

[1] There are many examples of this with all kind of licensing schemes. ECSIM pre-integrated tool is GNOME Evince.

[2] Again, there are many different text files viewer in every operating system. ECSIM pre-integrated tool is the GNOME text-editor "gedit".

© DEIMOS Space S.L., 2007

3.4. How to Install the Application

Provided that every pre-requisite is fulfilled, users can now proceed to install the application.

3.4.1. Framework installation

First, extract the framework zipped file into the desired location:

```
~/$ tar -xvf ecsim.<version>.tar.gz
```

This command will decompress all the files in the distribution and create a folder named "ECSIM" and inside it, the folder structure described in section 3.4.4. This folder will have to be the value of the $ECSIM_HOME variable defined in section 3.2.1

Go into the ECSIM folder and then, execute the installation script as follows:

```
~/ECSIM$ sh install <user> <password>
```

Where <user> is the name of a user in the MySQL database server with administrative privileges (typically, root) and <password> is its password. See also section 4.16.1.

This script will create the *ecsim* user into the database server and grant it with enough permission. This also will create and fill the nominal database information.

The installation script will interactively ask the user for values of the following environment variables:

Variable name	Purpose	Typical value
ECSIM_HOME	ECSIM system home folder	/home/<user>/ECSIM/
DEBUG_MODE	Presents debugging information to users	Off
TEST_HOME	ECSIM system test home folder	/home/<user>/ECSIM/tp
LD_LIBRARY_PATH	Folder location of the Intel Fortran Compiler libraries	/opt/intel/fc/9.1.036/lib:$LD_LIBRARY_PATH
PGPLOT_FONT	File describing supported PGPLOT fonts	/usr/local/pgplot/grfont.dat
OMP_NUM_THREADS	Default number of threads to use during execution of models using the OpenMP (multi-processing) module.	2
KMP_STACKSIZE	Sets the stack size for each thread created for models using the OpenMP (multi-processing) module.	512M

[3] In almost every Linux system, the "diff" program is installed by default.

© DEIMOS Space S.L., 2007

MSI_LOOKUP	Folder location of auxiliary MSI lookup tables	/home/<user>/ECSIM/models/aux/msi_lookup/
SCATT_LIB	Folder location of auxiliary scattering libraries	/home/<user>/ECSIM/models/aux/scattering_libraries/
STD_ATMOS	Folder location of auxiliary standard atmospheric profiles	/home/<user>/ECSIM/models/aux/standard_atmos_profiles/
SURF_FILES	Folder location of auxiliary surface definition files	/home/<user>/ECSIM/models/aux/surface_files/

Once you finished extracting the platform, it is necessary to copy in ECSIM_HOME folder a file from the MySQL Connector/J package: mysql-connector-java-5.0.4-bin.jar. This file is needed for the system to access the database server.

If you want to use another version of the MySQL Connector/J you can also copy it into the ECSIM_HOME folder but then, you must edit the "ecsim" running script to include the new jar file. For example, if version 5.1.6 wants to be used, the next line must be changed:

```
java -cp mysql-connector-java-5.0.4-bin.jar:bin: ECSIM $*
```

by this one:

```
java -cp mysql-connector-java-5.1.6-bin.jar:bin: ECSIM $*
```

3.4.2. Scattering libraries and DEM installation

3.4.2.1. Scattering libraries

The scattering libraries are an essential component of ECSIM. The scene (UFF) files do not contain any information about the optical properties of the clouds and aerosols. To store this information at each grid point for all the necessary wavelengths for an end-to-end simulation would be infeasible. Instead the UFF files only contain information on the number and sizes of the aerosol and cloud particles present at each grid point. The necessary optical information (i.e. extinction coefficient, single-scattering albedo, phase function etc..) are stored in pre-computed scattering libraries.

A scene can only be created or used as input by other ECSIM applications if all scattering libraries it references have been installed on the system. One can check which libraries are referenced by a UFF file by inspecting the associated header file using any ASCII viewer. The "scattering list" files are listed near the top (line 4 onwards) of the header files.

Your ECSIM distribution already has a minimal set of scattering libraries for running the included scenes. In case you need other files not included in the provided set, a more extensive set may be built in-situ by the user by running the following steps.

1. 1. cd $ECSIM_HOME/models/aux/scattering_libraries
2. 2. unzip scatt_libs.zip
3. 3. cd data_files
4. 4. Open the file "build_mie_tables.csh" with a text editor.
5. 5. Change the shell variables defining the compiler and compiler options to suit the target system (default is ifort and -O3) and save the files.

© DEIMOS Space S.L., 2007

6. 6. Execute 'source build_mie_tables.csh >& LOG'

The script will then compile a number of Mie scattering codes and then execute various other scripts in order to build the full set of libraries.

Please note that this process can be extremely long (around twelve hours in a machine matching the hardware pre-requisites).

3.4.2.2. DEM

ECSIM uses a DEM called GETASSE30, that stands for Global Earth Topography And Sea Surface Elevation at 30 arcsec resolution. This DEM (or Digital Elevation Model) is a combination of four datasets (the SRTM30, ACE, Mean Sea Surface data and EGM96 ellipsoid), being eventually referenced to the WGS84 ellipsoid. The file is available at an ESA server as binary tiles with 1800*1800 points (a squared area 15x15 degrees size). These files must be placed in a specific folder that you must create with:

```
~/ECSIM$ mkdir models/orbit_dms/aux/dem_data
```

Alternatively, you can create a link to the place where you have downloaded them:

```
~/ECSIM$ cd models/orbit_dms/aux/
~/ECSIM$ ln -sf <original_dem_folder> dem_data
```

3.4.3. Model compilation

Once the framework is successfully installed, locate the $ECSIM_HOME/models folder and once into it, edit the "Makefile.in" file.

```
~/ECSIM/models$ vi Makefile.in
```

Inside you must locate and edit the following variables and match them with your actual folder structure:

- NC_INCLUDE. Location of the NetCDF include-files.
- NC_LIB. Location of the NetCDF library.
- PGPLOT_LIB. Location of the PGPLOT library.
- GCC_LIBS. Your GNU C Compiler libraries location.

Another variable you can set is the one under section "PGPLOT compile flags", PGPLOT_FFLAGS. This variable must be adapted to your PGPLOT library installation process. It depends on what Fortran compiler you used and the drivers you included. Example values can be these:

- PGPLOT_FFLAGS=-lifport $(PGPLOT_LIB) -lX11 -L$(GCC_LIB) -lgcc -lgfortran -g -traceback –CB (Library compiled with gfortran)
- PGPLOT_FFLAGS=-lifport $(PGPLOT_LIB) -lX11 -L$(GCC_LIB) -lgcc -lm -lc -lg2c -g -traceback –CB (Library compiled with g77)
- PGPLOT_FFLAGS=-lifport -lpng $(PGPLOT_LIB) -lX11 -L$(GCC_LIB) -lgcc -lm -lc -traceback –CB (Library compiled with ifort, and including PNG driver)

Before the models are compiled, you must copy your CFI installation folder to $ECSIM_HOME/models/orbit_dms/aux/CFI.

© DEIMOS Space S.L., 2007

Then, you can compile the scientific models and tools with this sentence.

```
~/ECSIM/models$ make all
```

3.4.4. Folder structure

This section provides a general description of the ECSIM folder structure and its contents.

Folder name (indented)	Contents
❑ .	ECSIM home root ❑ "install". Scripts for setup in Linux. ❑ "ecsim". Starting-up script for Linux. ❑ "ECSIM.properties". The configuration file. ❑ "ECSIMdb.sql". The original MySQL database script. ❑ "lgpl-3.0". Licensing scheme file
❑ aux	Auxiliary files for the framework
❑ bin	Binary files of the framework.
❑ doc	Framework documentation
❑ models	ECSIM models ❑ "Makefile" ❑ "Makefile.in"
❑ aux	Auxiliary files for the models
❑ channel_definitions	Channel definitions.
❑ msi_lookup	Lookup tables with scene reflectivity values. They are intended to speed up the retrievals based on MSI L1 data.
❑ scattering_libraries	Scattering libraries.
❑ solar_data	Solar data
❑ standard_atmos_profiles	Standard atmospheric profiles
❑ surface_files	Surface data for the scene description
❑ xml	Fortran XML libraries
❑ common	Common sources, modules and utilities used by the models
❑ <model_name>	Model root. Every model has one directory structured as this one. Model folders might include additional folders.
❑ aux	Auxiliary files
❑ bin	Binary files
❑ conf	Configuration files
❑ src	Sources
❑ TEST_DMS	ECSIM models test folder. Scripts and file references for the model unit tests.

© DEIMOS Space S.L., 2007

Folder name (indented)	Contents
❑ products	Some product file samples
❑ scenes	Some scene files samples (xml and uff files)
❑ sessions	ECSIM sessions root folder
❑ <session_name>	Session folder. Every session, once executed, has one directory structured as this one. This folder will have the session script and, if generated, the session report.
❑ <index.simulation_name>	Simulation folder. Every simulation into the session has a directory structured as this one. This folder will have the input and configuration files used by the models and their outputs.
❑ src	Framework sources
❑ tmp	Framework temporal directory
❑ tools	ECSIM tools root folder
❑ tp	Test procedures home folder. Every test run in the system test campaign has its own sub-folder structure.
❑ tp-<tp_number>	Individual test procedure folder. Each test is identified with a four digits number.
❑ input	Input files needed for the test.
❑ output	Generated output files during the test.
❑ reference	Reference data needed for the test.

3.4.5. Licensing scheme

Due to the heterogeneous precedence of the contributions to this ECSIM project, various licensing schemes are applied. Here readers can find a list of the different license types used in the project.

Item	License type	Comment
Models (developed by KNMI)	BSD LGPL	Exceptions: 1) cloud fractal generator (model type "scene") is GPL 2) fast lidar scattering model (model type "forward") is GPL
Orbital model	[ESA]	This is based on the ESA-CFI only available upon request (to ESA). Licensing is then conditioned to ESA.
Framework (HMI)	LGPL	

© DEIMOS Space S.L., 2007

3.5. How to Start the Application

The ECSIM framework can be launched using a command line interface and executing the following command:

```
~/ECSIM$ sh ecsim
```

or into a Bourne shell.

```
~/ECSIM$ ecsim
```

If the ECSIM[4] is launched with no parameters the GUI will show up normally. This behaviour can be modified providing the following parameters:

- *-noGUI*. No graphical user interface is shown. Running ECSIM with this option enables experienced users to go faster in the execution of their sessions. This option only makes sense when used with the following one.
- *-execute <session_identifier>*. The ECSIM framework will launch the execution of a previously defined session with the defined parameter values. This option can be used with or without the previous one.

  The session identifier shall correspond to the one stored into the ECSIM database.

  Author and description parameters are optional and will be included in the session execution general data.

  Fields can contain more than one word if they are enclosed in singles quotes.

An example to illustrate the execution in batch mode of a "Radar" simulation is as follows:

```
~/ECSIM$ ./ecsim -noGUI -execute Radar DMS Test_radar_simulation
```

This sentence will execute the ECSIM in text mode (no graphical interface), create a session named "radar", created by the author "DMS" and described as "Test_radar_simulation". Then it will execute it, intercepts all the events generated and stores in the database its results. Then, the system will stop.

3.5.1. Environment variables

Once you have successfully run for the first time your ECSIM installation you would like to go to the "System Configuration" module and adapt the default environment variables to match your distribution. This is described in detail in section 4.5.1

[4] It is assumed here that the environment variable PATH contains the path to the java virtual machine executable.

© DEIMOS Space S.L., 2007

4. ECSIM – REFERENCE MANUAL

This section provides a detailed description of all the elements that conforms the ECSIM graphical user-interaction.

4.1. Main window

Here the look-and-feel, operational behaviour and design features common to the ECSIM GUI (in advance, the MMI, Man Machine Interaction) are presented.

The MMI accepts input via devices such as the computer keyboard and mouse and provides articulated graphical output on the display. Thus certain aspects of the MMI implement also the **Object Oriented Interface (OOUI**) paradigm because it is built from different pieces, or objects with several properties and operations.

The ECSIM MMI also follows the **Multiple Document Interface (MDI)** pattern. This approach has been chosen because of its flexibility, as it lets users organize the layout of the information as desired, showing only relevant windows and in the way users want.

The MDI pattern consists of a "parent" container that can host inside several "internal frames". These internal frames are intended to present independent modules of the simulator. For example, each time the user wishes to perform operations with the list of models of the system, a "model manager" frame will pop-up inside the bounds of the main window listing the list of models currently available within ECSIM.

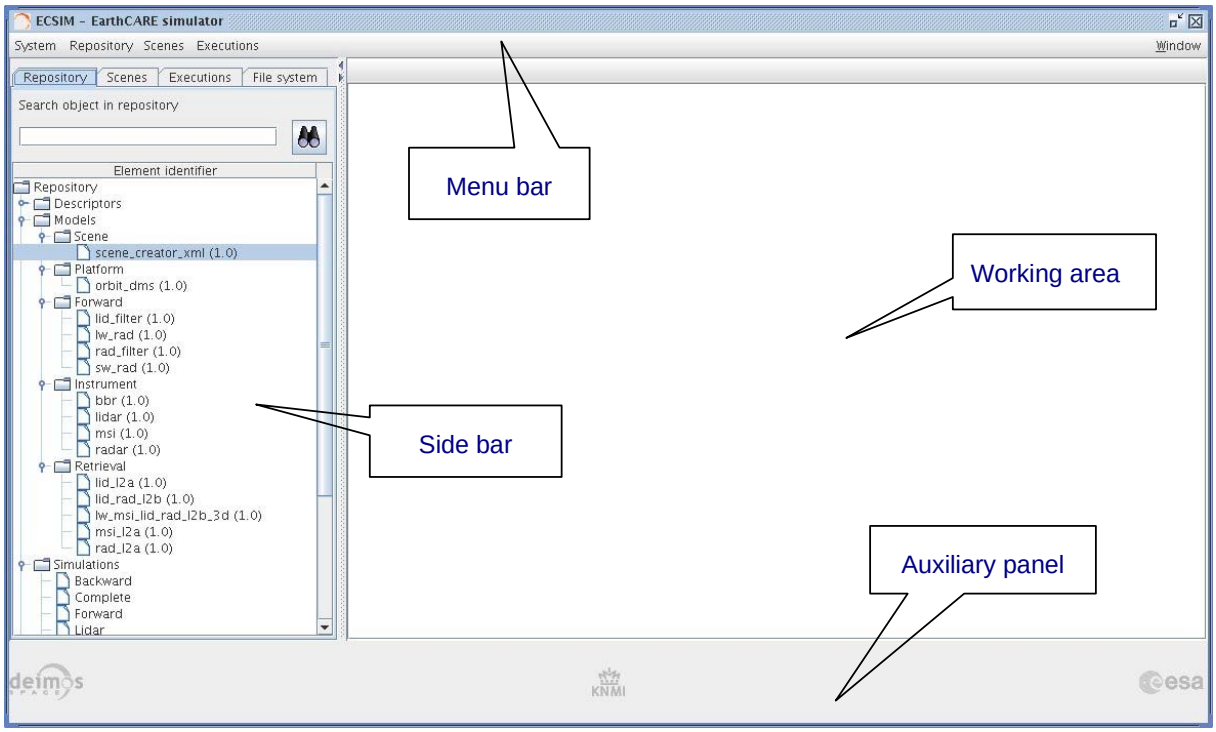


Figure 4-1: Main window appearance

© DEIMOS Space S.L., 2007

All the windows have common operations to help their usability: main window, internal frames or dialogues can be closed, resized, maximized or minimized to fit the user's needs.

This main window shown in Figure 4-1 includes a menu bar to provide keyboard and mouse access to the simulator main functions as well as functions regarding frames management and application basis.

The MDI pattern provides some useful capabilities to arrange internal frames (e.g. Figure 4-2) appropriately like cascading or tiling them. Also internal frames can be "iconized" to give more available space. When a user iconizes a frame it can be restored by clicking the button with its name in the "available frames" tool bar or in the corresponding menu item at the "windows" menu.

Occupying the central and main region there is a working area. This area is where all internal frames are going to be created and main interaction is held. Besides that, this working area implements a "scrollable" panel in order to easily navigate through frames surpassing its bounds.

At the left side of the working area there is a system objects navigator, a "side bar" aiming to provide a quick access method to every item known by the system: repository of descriptors, models, simulations and sessions; the list of simulation execution results; a view to the scenes folder and also a file system browser to navigate through the contents of the application's directory.

The main window's footer area shows the corporate logos of the companies in charge of the development of ECSIM (DEIMOS and KNMI) as well as the ESA logo.

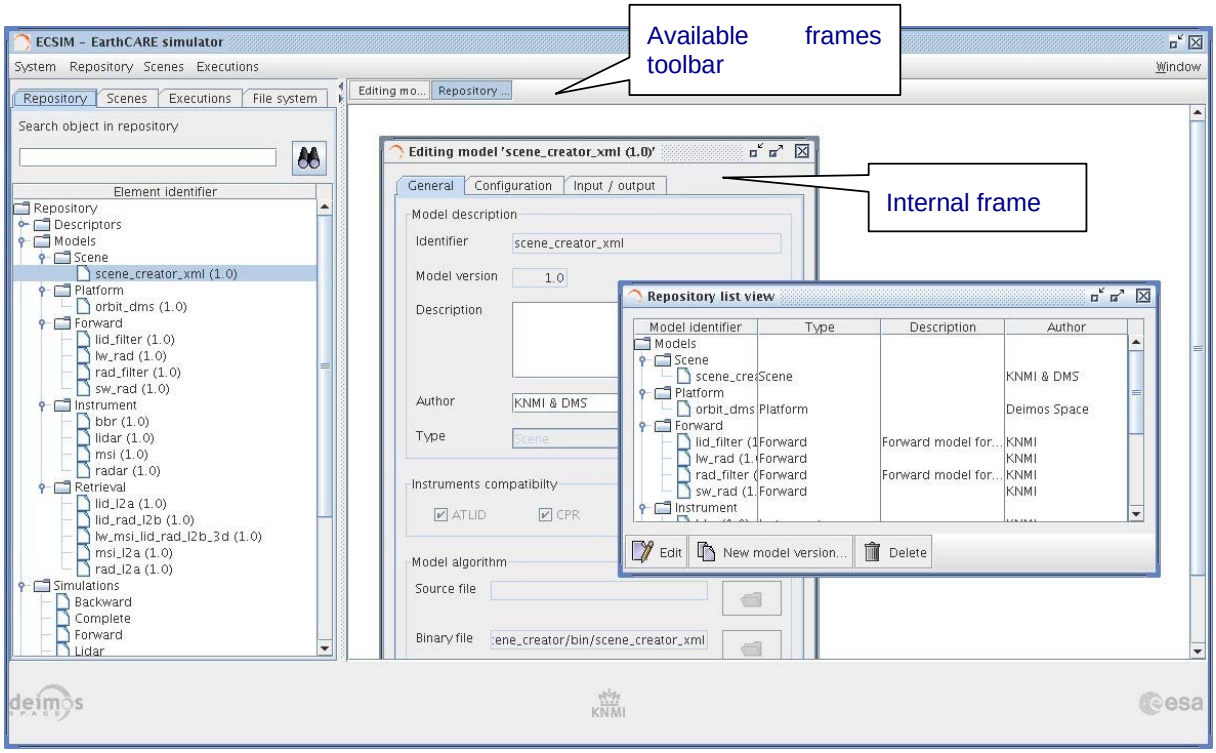


Figure 4-2: Main window appearance showing internal frames and scroll bars

The MMI provides a menu bar (Figure 4-3) at the upper side of the main frame to show some capabilities of the system.

© DEIMOS Space S.L., 2007

Figure 4-3: Detail of main menu bar

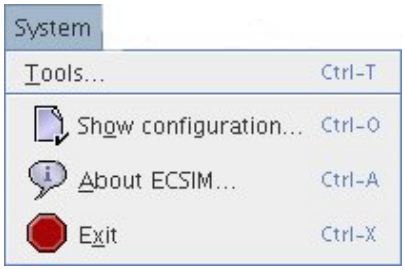


Figure 4-4: Detail of a menu, showing menu items

It is shown in Figure 4-4 that a menu item is an icon with graphically describes the function, the name of the function and a "quick access" key combination. Users can quickly access this functionality pressing this key combination or the letter underscored in the function name while the menu is rolled down.

There are also some contextual or "pop-up" menus that users can access by clicking the right button of the mouse while over certain controls. These "pop-up" menus have the same appearance as menus rolling down the menu bar.

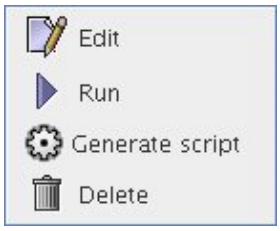


Figure 4-5: Detail of a contextual menu

It can be seen in Figure 4-5 that a pop-up menu acts exactly like a menu at the main frame. They also provide mouse and keyboard access to certain capabilities.

4.2. Generic Functions, dialogues and displays

This section is meant to describe the design of generic functions, dialogs and displays used by the MMI.

There are some functionalities of the MMI that show a "file chooser" dialog as shown in Figure 4-6.

© DEIMOS Space S.L., 2007

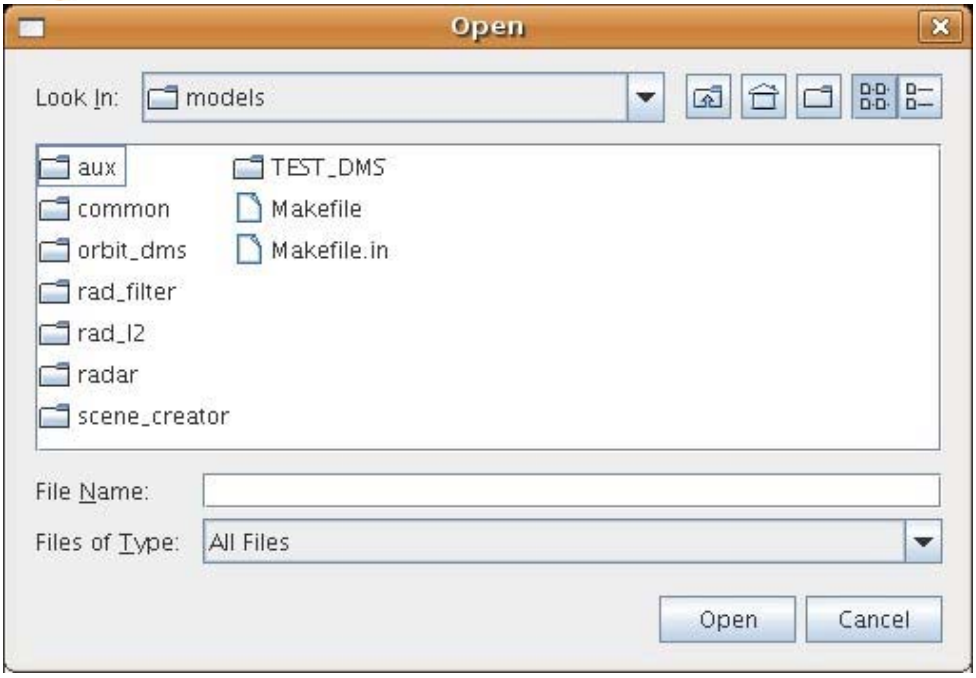


Figure 4-6: File chooser dialog

This dialog helps the user to browse the system directory to select a certain file or list of files. It provides some sorting, filtering and file operations very common and already known for the majority of the users.

Throughout the ECSIM MMI some functionality could show information to the user and could ask for some input in response to an answer. The MMI will present some "modal" dialogs that will get the system focus until the user provides an answer. These modal dialogs will block the input to other areas of the application until a response is given.

Figure 4-7: Dialog example

These dialogs will typically provide a message with an "OK" button or give a yes-or-no question or another question with different options. The dialogs will provide information with a clear description of the event.

4.3. Frame management

Accessing to the "Window" menu of the main menu bar you can find all the functionality provided for the frame management. Figure 4-8 shows that this menu has the common aspect of other MDI applications.

© DEIMOS Space S.L., 2007

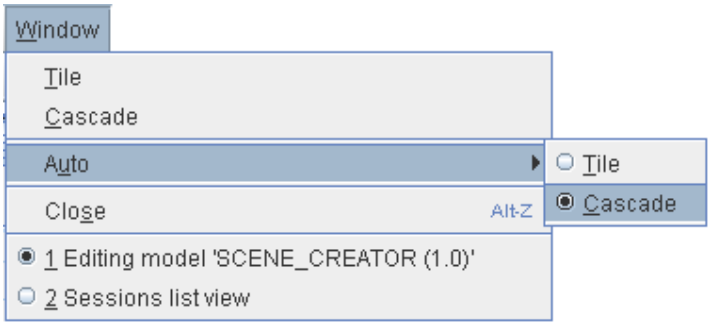


Figure 4-8: Frame management menu

This is a brief description of these well-known capabilities:

- Tile: arrange internal frames as tiles over the working area
- Cascade: arrange internal frames ordering them as in a cascade.
- Auto: presets an arranging style for newly created frames.
- Close: hides and destroys the active frame.

Then, the list of open frames is shown in the rest of the menu. Active frame is pointed with a filled round-button.

Other frame management functionalities can be found in the header of every internal frame and in the main frame (not in dialogues).

Figure 4-9: Internal frame header

Note in Figure 4-9 three little icons at the right border of the header. Those are the "minimize", "maximize"/"restore" and "close" operations.

If the user minimizes a frame it disappears from the working area but it can be restored from the "available frames toolbar"

4.4. Side bar

On the left side of the main frame there is side bar, grouping different views of the ECSIM areas: Repository, Scenes, Executions, and File system.

As you can see at the right-upper corner of Figure 4-9, two little triangles can be touched to minimize or maximize the width of the side bar. The dotted bar can also be dragged to dynamically change its width.

4.5. System

Selecting the "System" menu the users can access to the following functionalities:

© DEIMOS Space S.L., 2007

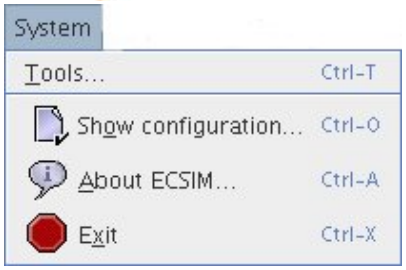


Figure 4-10: System menu

These functionalities are given to control the general characteristics of the whole ECSIM system.

Elements are structured into the repository first by element type (descriptors, models, simulations and sessions) and then by family.

The repository will show only the elements containing a given substring of characters written in this text box.

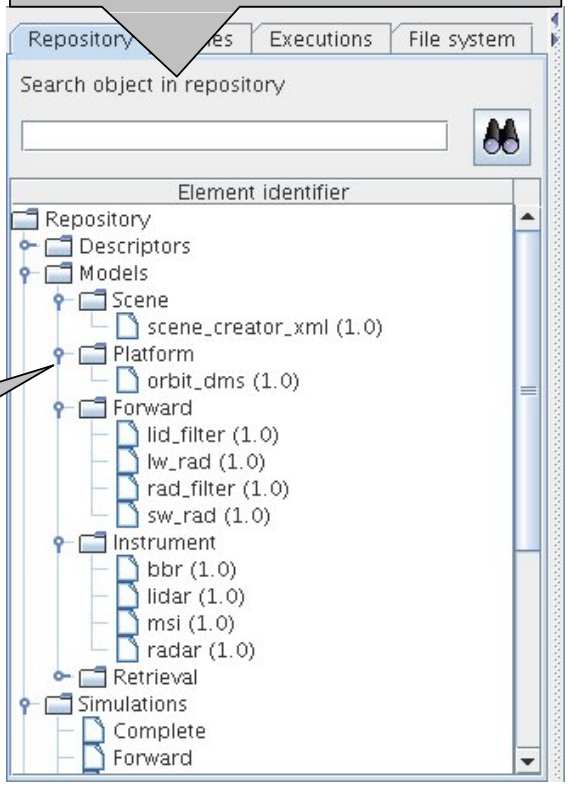


Figure 4-11: Side bar

© DEIMOS Space S.L., 2007

4.5.1. Show configuration

Selecting this menu option, the dialog shown in Figure 4-12 will show up.

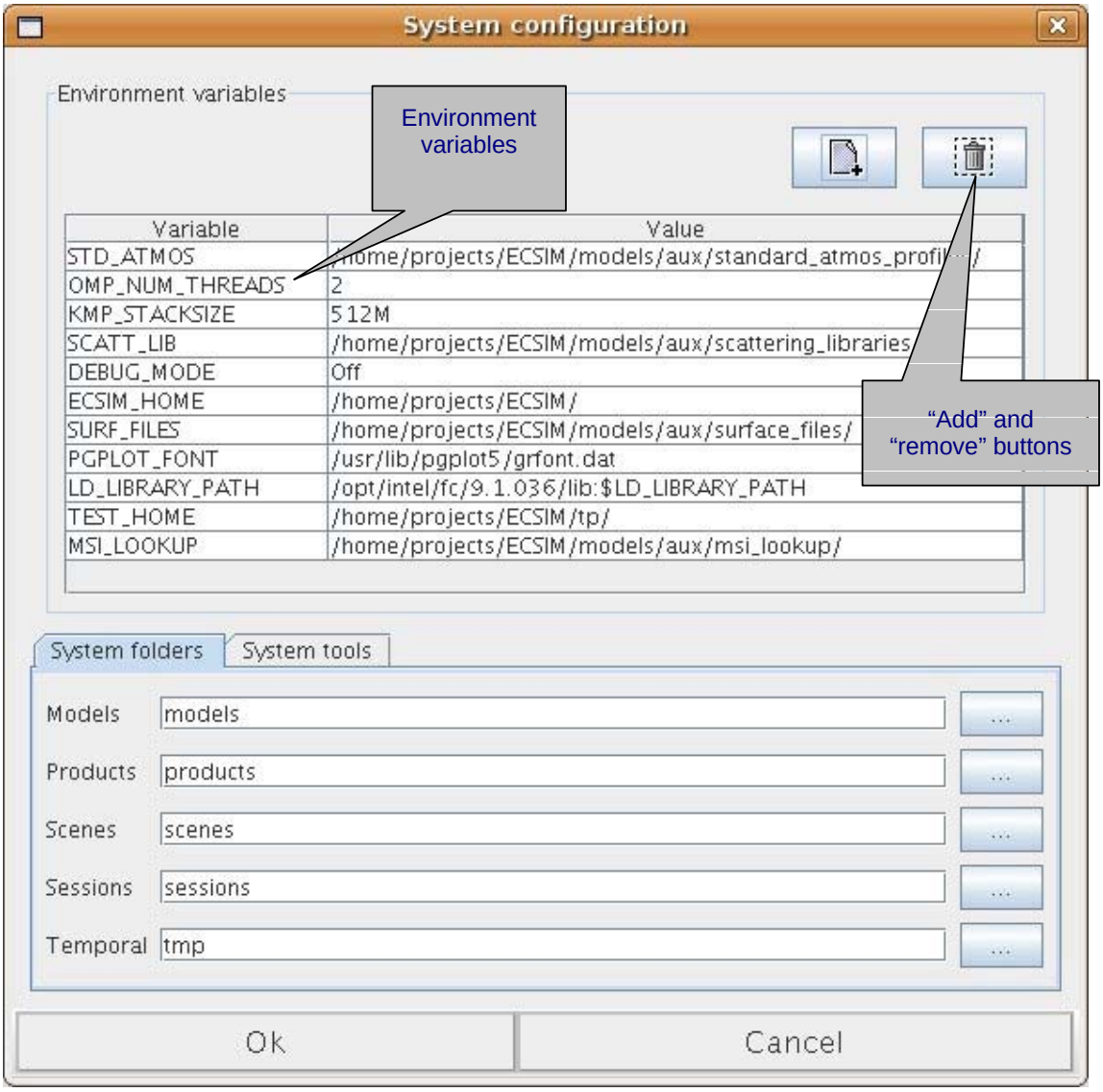


Figure 4-12: System configuration

In this dialog users can modify the following characteristics of the system:

- **Environment variables**. A list of environment variables and associated values are shown in this table. Once a model or a product tool is being executed, they can access these variables if they need them because the system makes them available to the execution process. Users can "add" or "remove" an environment variable using the given buttons. Double-clicking on an already existing variable the user can edit its name and value.

  One mandatory variable is $ECSIM_HOME (as explained previously). This points to the ECSIM base folder location.

© DEIMOS Space S.L., 2007

There is an environment variable called "DEBUG_MODE" that controls the verbosity of some model executions. Setting to "On" or "Off" can enable or disable this functionality.

- **System folders**. Users can change the default locations for these directories:
  - *Models*. Here is where the model packages can be found. These packages are just a folder-structure containing the needed elements for the model to be successfully run following the directives given in [RD ICD].
  - *Products*. This is the best place to store product files of common use.
  - *Scenes*. The set of XML files inside this directory are going to be scanned and those representing valid scene descriptions are going to be listed in the "scenes" tab of the side bar.
  - *Sessions*. This is the place where all the files associated to session executions can be found. Execution scripts, report files and, by default, configuration and output files generated are going to be stored here following the [RD ICD] directives.
  - *Temporal*. Some intermediate files are going to be stored temporarily in this directory.
- **System tools**. There is a list of system tools that must be configured in order to activate some scene functionalities.

All around this configuration dialog and the rest of the ECSIM system (except those places where something else is specified), the user can choose to input the absolute path of a certain file or folder or the path relative to $ECSIM_HOME.

© DEIMOS Space S.L., 2007

4.5.2. Tools

As explained in section 1.3, a tool is an external executable file that performs a given action to a certain group of files. Used as part of the ECSIM framework and associated to a certain file extension, these tools can be called to perform off-line operations to products involved in simulations.

Tools are definable by the user.

Accessing to this functionality, the user is able to manage the ECSIM product tools.

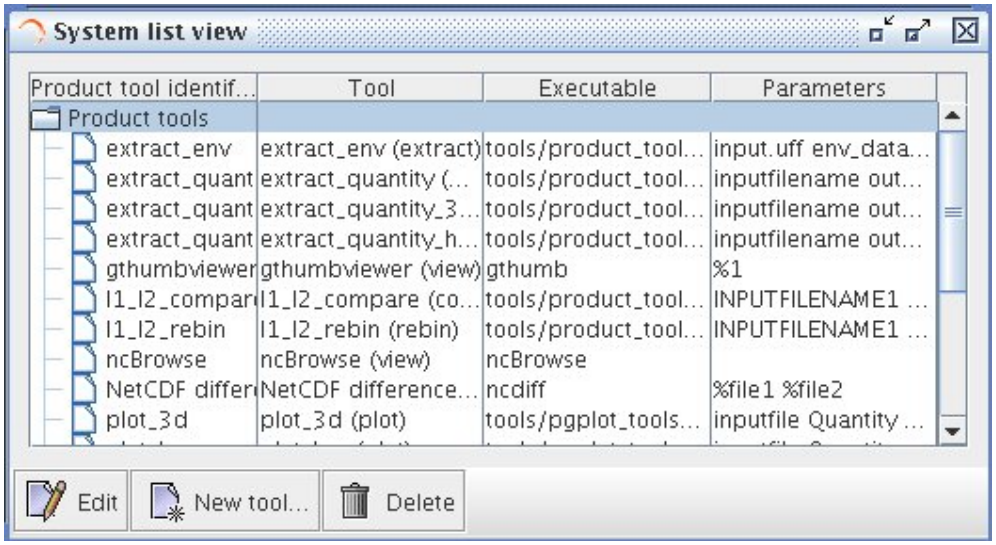


Figure 4-13: Tools list view

A list of tools, showing its identifier, action, executable and parameters is given.

Tools are definable by the user. Thus, new tools can be added by clicking the "New tool" button.

4.5.2.1. New tool

Accessing to this functionality, a new window appears to let the user create a new tool.

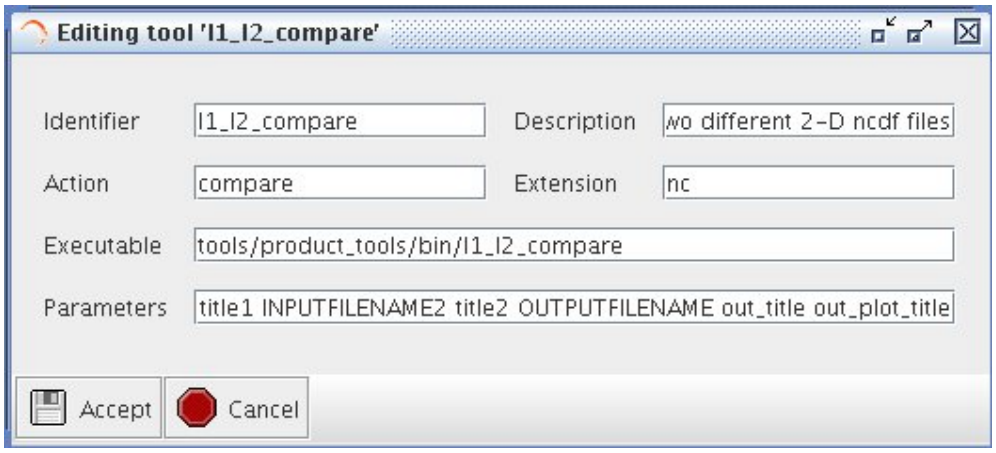


Figure 4-14: Tool. Creation

© DEIMOS Space S.L., 2007

Users can define the following attributes:

Attribute name	Format	Purpose	Sample
Identifier	Medium string	This is a unique identifier of the tool.	"XML editor"
Description	Long string	A brief description of what this tool will do and need.	"This tool will open an XML file for edition"
Action	Medium string	What the tool is going to do.	"edit"
Extension	Short string.	The type of files that this tool is going to be applied to.	"xml"
Executable	Long string	Location of the binary file that is going to be called to execute the product.	"gedit"
Parameters	Long string	The list of parameters that will follow the executable. No variables can be passed from the HMI.	"-f $file1"

4.5.2.2. Edit tool

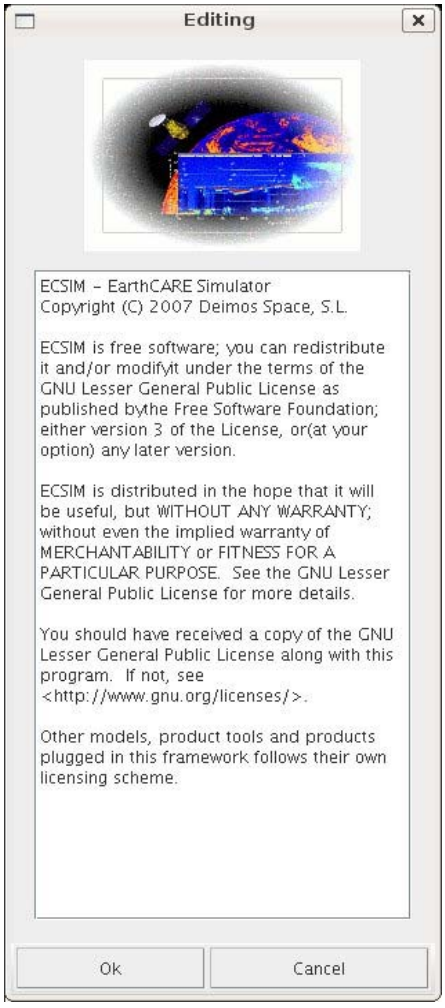
Selecting this functionality, the user can access and edit all the attributes of the selected tool.

4.5.2.3. Delete tool

Selecting this functionality, the user can delete the selected tool.

4.5.3. About ECSIM

Accessing this functionality, the system will show a dialog with the copyright and license scheme for the ECSIM platform.


Editing
ECSIM – EarthCARE Simulator
Copyright (C) 2007 Deimos Space, S.L.
ECSIM is free software; you can redistribute it and/or modifyit under the terms of the GNU Lesser General Public License as published bythe Free Software Foundation; either version 3 of the License, or(at your option) any later version.
ECSIM is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.
You should have received a copy of the GNU Lesser General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.
Other models, product tools and products plugged in this framework follows their own licensing scheme.
Ok
Cancel

4.5.4. Exit the system

Upon the selection of this function, ECSIM will ask to stop every internal process and will end its execution. This is the recommended way of ending the application.

© DEIMOS Space S.L., 2007

4.6. Repository

The ECSIM system can be divided in three logical parts: repository, scenes and executions.

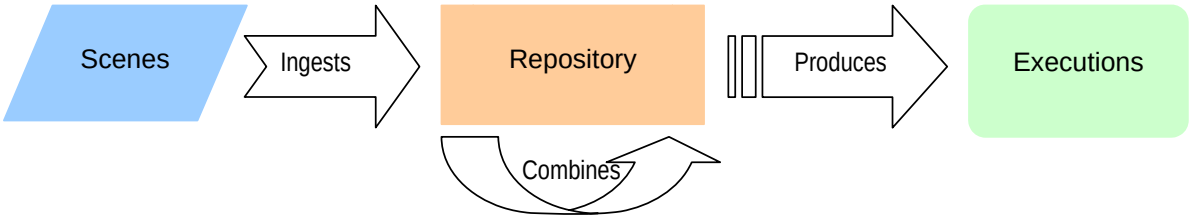


Figure 4-15: ECSIM logical flow

The first of these parts, repository, is involved in defining the "static" view of the system, the definition of the ECSIM tool. This is the place to access and define all the elements that, later on, will serve as bricks for the simulation executions:

- **Models** – executable entities that will perform the scientific or engineer calculations;
- **Descriptor** – to define which models can be linked in which way;
- **Simulations** – sequences of models linked by the descriptors;
- **Sessions** – complex set of simulations with added input/output/configuration files and product tools.

Into the repository tab of the side bar users can find a tree-like structure containing the definitions for all known models, descriptors, simulations and sessions.

This tree-like structure can be collapsed or expanded double-clicking in the group name or clicking in the "anchor" icon.

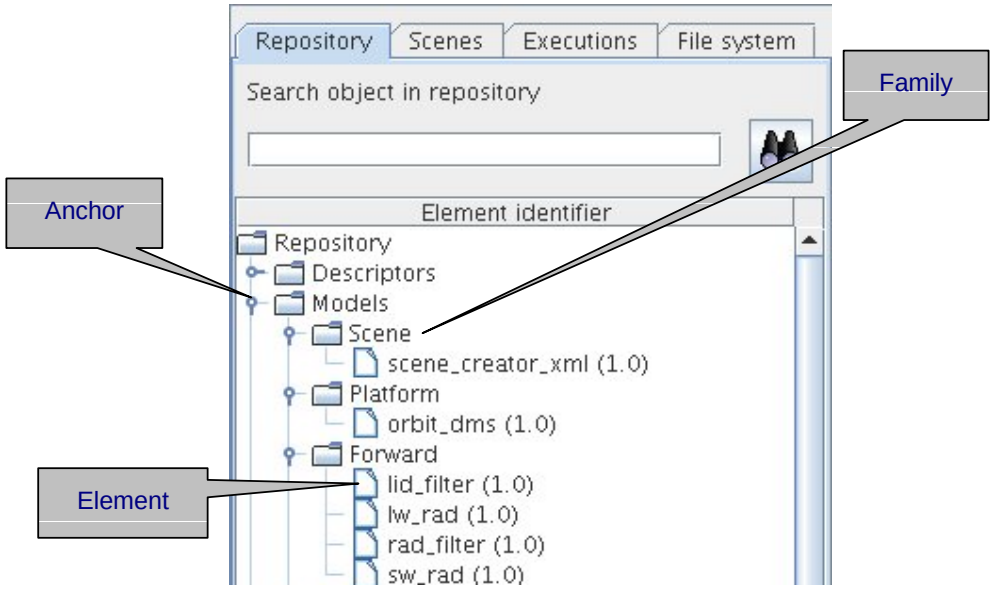


Figure 4-16: Repository view

Every row marked with a "folder" icon represents a group of elements. Thus, into the "models" group, the model families can be found, i.e. Scene, Platform, Forward, Instrument and Retrieval.

© DEIMOS Space S.L., 2007

Every row marked with a "document" icon represents an element definition of the repository. Right-clicking over them, a menu pops up containing some associated commands. These menus are context-sensitive, meaning that different types of elements have their own associated commands. These commands are going to be explained in detail in each element's section.

A double click over the elements will activate the first associated command of the menu (typically, edition).

4.7. Descriptors

ECSIM has the possibility to define the set of input and output files (called descriptors) used to connect different models in simulation runs.

Users can access the list of nominal descriptors (those provided in the default distribution) inside the repository view of the side bar, as seen in Figure 4-17.

Accessing the corresponding menu of the main menu bar or the context-menu of the side bar, users can activate the following functionalities:

- *List* – presents the list of existing descriptors;
- *Creation* – creates a new descriptor into the system;
- *Edition* – edits an existing descriptor to enter changes;
- *Deletion* – deletes a descriptor from the system.

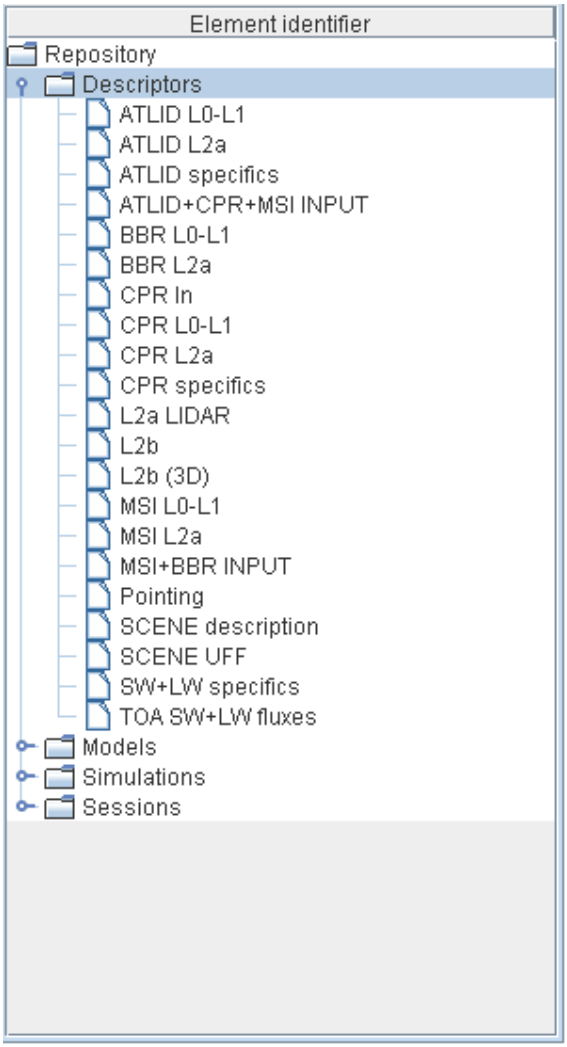


Figure 4-17: Descriptor. Side bar

4.7.1. Descriptor list

Users can access to a window that provides a tree-like structure with the list of descriptors known by the system, just as in the side bar but with the additional information of its identifier and its description.

© DEIMOS Space S.L., 2007

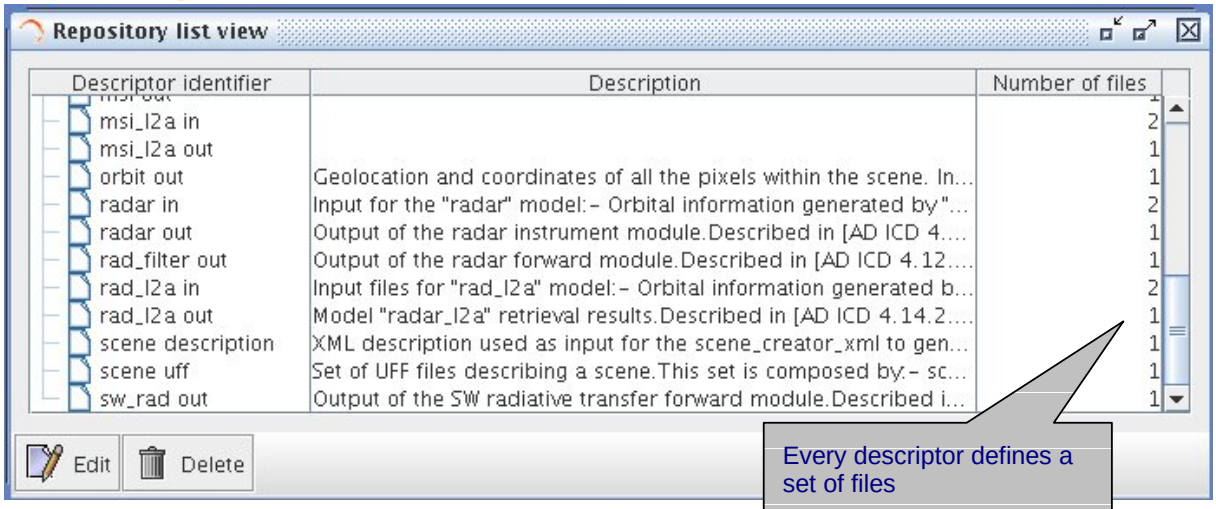

Repository list view
Descriptor identifier
Description
Number of files
msi_l2a in
msi_l2a out
orbit out
Geolocation and coordinates of all the pixels within the scene. In...
radar in
Input for the "radar" model:- Orbital information generated by"...
radar out
Output of the radar instrument module.Described in [AD ICD 4....
rad_filter out
Output of the radar forward module.Described in [AD ICD 4.12....
rad_l2a in
Input files for "rad_l2a" model:- Orbital information generated b...
rad_l2a out
Model "radar_l2a" retrieval results.Described in [AD ICD 4.14.2....
scene description
XML description used as input for the scene_creator_xml to gen...
scene uff
Set of UFF files describing a scene.This set is composed by:- sc...
sw_rad out
Output of the SW radiative transfer forward module.Described i...
Edit
Delete
Every descriptor defines a set of files

Figure 4-18: Descriptor list view

4.7.2. Descriptor creation

Users can define new descriptors in case they want to accommodate third-party models that cannot make use of any of the nominal descriptors.

The frame shown in Figure 4-19 is responsible to define the descriptor's characteristics:

Attribute name	Format	Purpose	Sample
Identifier	Medium string	Descriptor's unique name.	"LIDAR In"
Description	Long string	A brief description of its composition or the purpose of the set of files.	"Orbit information and radiative transfer information"

It is possible to alter the set of files that integrates the descriptor. Users can edit, add or remove files. An individual file must be described by these two parameters:

Attribute name	Format	Purpose	Sample
Default file	Medium string	The default location and name of the file. This is the file that is going to be suggested during the session definition (see section 4.10. Sessions).	"orbit.xml"
Description	Long string	Brief description of the file's composition, its purpose or its type (XML, UFF, NETCDF, etc.).	"XML file with Orbital information"

It is important to note that **the default file name is the way to know if two models are compatible and to connect them in the simulation definition**. More information can be found in section 4.9.2.2.

The order that those files occupy in the list is important. This order must fulfil the directives of [RD ICD] or the command line specification of the third-party model because how you sort the files will define the

© DEIMOS Space S.L., 2007

order of the input and output files in the command line of the model execution. This order can be altered with the "up" and "down" buttons that move the selected file through the list.

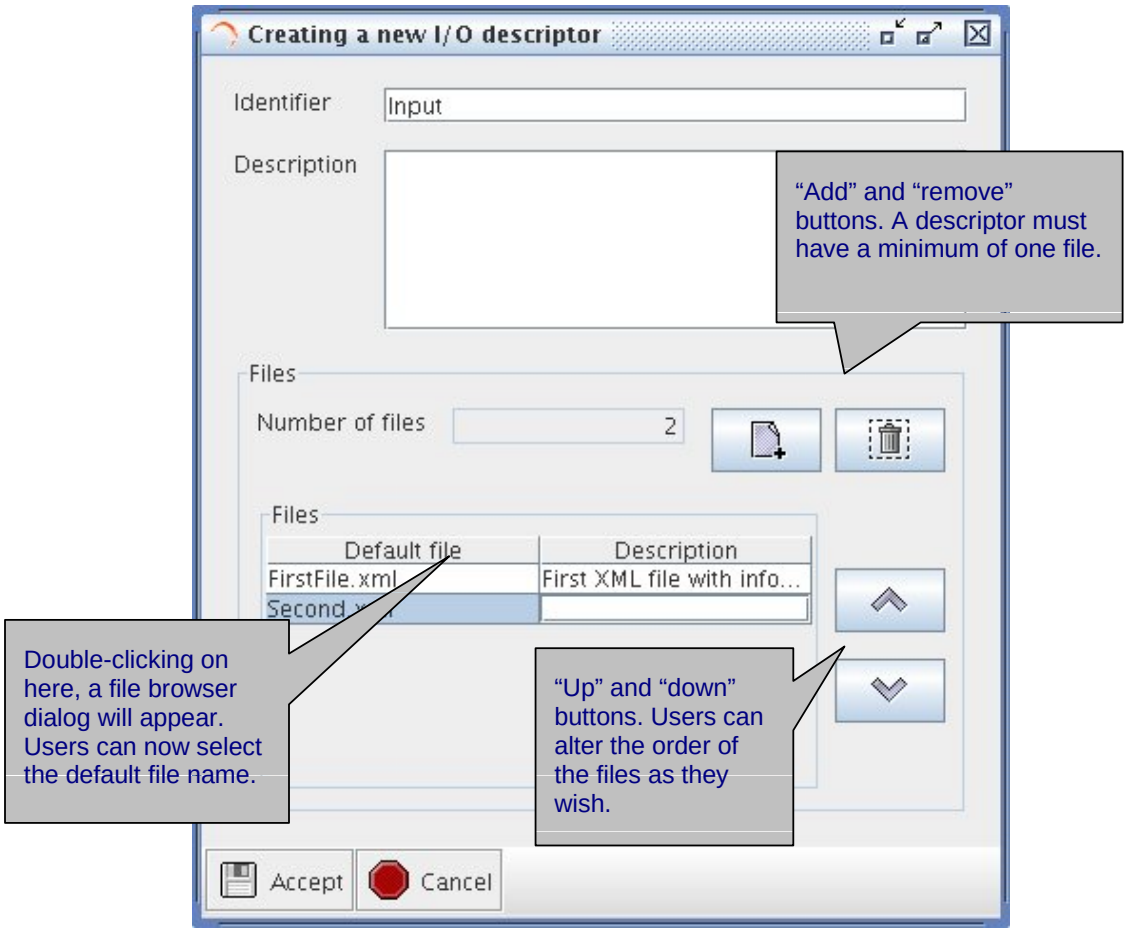

Creating a new I/O descriptor
Identifier
Input
Description
"Add" and "remove" buttons. A descriptor must have a minimum of one file.
Files
Number of files
2
Files
Default file
Description
FirstFile.xml
First XML file with info...
Double-clicking on here, a file browser dialog will appear. Users can now select the default file name.
"Up" and "down" buttons. Users can alter the order of the files as they wish.
Accept
Cancel

Figure 4-19: Descriptors. New descriptor

4.7.3. Descriptor edition

Users can select a certain descriptor and choose the option to edit it. Note that for consistency reasons only the description is allowed to be changed.

4.7.4. Descriptor deletion

Users can also select a descriptor to delete it. Once users confirm the operation the descriptor is erased from the repository. **Note that for consistency purposes, every model, simulation, session (and its results) that makes use of this descriptor will also be deleted from the system**.

© DEIMOS Space S.L., 2007

4.8. Models

According to the definition given in section 1.3, a model is an executable entity that can take part in a simulation.

Users are able to manage all models that can take part in ECSIM simulations. The operations upon models are:

- List – present the list of existing models;
- Creation – capability to create a new model into the system;
- New version – create a new version of an existing model;
- Edition – edit an existing model to enter changes;
- Deletion – delete a model from the system.

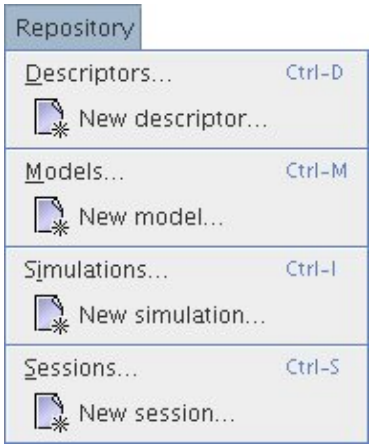


Figure 4-20: Repository menu

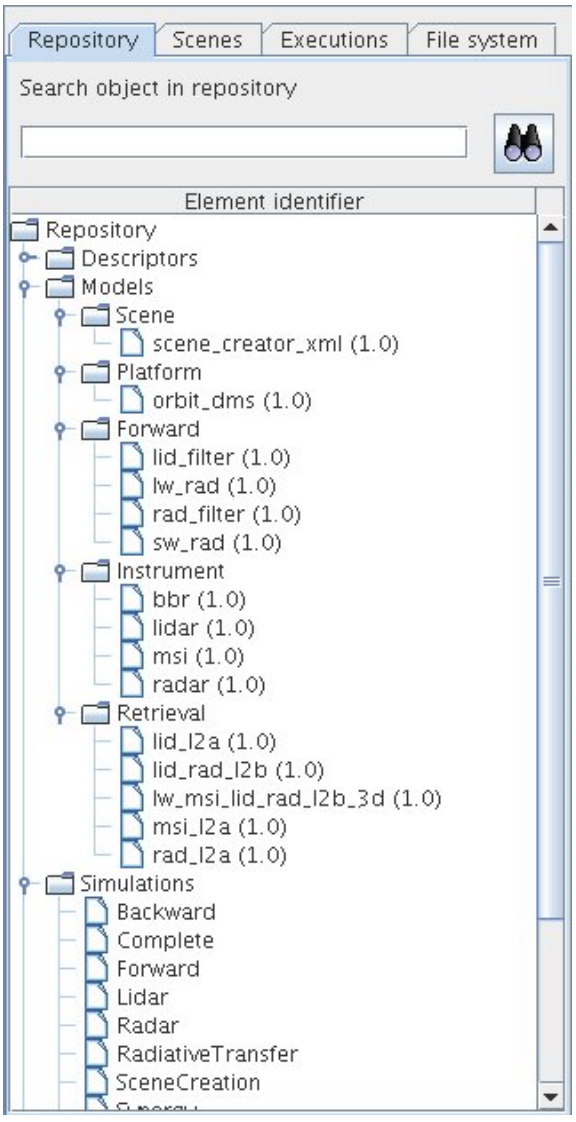


Figure: Models. Repository view

Users can access to some of these operations at the models menu in the menu bar of the main window (Figure 4-20) or in the correspondent context-menu of the repository view.

Figure 4-21: Model. Pop-up menu

© DEIMOS Space S.L., 2007

4.8.1. Model developers guideline

ECSIM can integrate as a model almost every executable piece of code that follows the requirements described in [RD ICD]. Given this situation, the model can be integrated and executed into the system. Nevertheless, model developers must have in mind the following points:

- Memory handling is responsibility of the model. ECSIM does not manage memory assignments and does not destroy any data structure created by the model;
- A model can create child processes, but their management is still on the model developer's side;
- ECSIM system does not detect when a model execution is "halted" or in an infinite loop. Please send some logging information (see [RD ICD]) to the ECSIM every two seconds (at most) to let the user know there is no problem;
- Execution performance of the model could be slightly slowed because of the messaging interception.

4.8.2. Model list

Accessing to this functionality from the main menu or from the repository, the system will show a list of models known by the system. Figure 4-22 shows an example of the window that appears upon its selection.

As models and their versions follow a hierarchical structure, i.e. a single model belongs to a family of models, the information is organized as a tree-table, which put hierarchical data in columns, like a table, but uses an indented outline structure in the first column to illustrate the tree structure.

Users can select a certain version of a model and perform the operations shown in the toolbar.

Data attributes shown in this tree-table are model ID, version number, type, description and the name of the author.

At the "model identifier" column, it is presented the indented outline structure with folder icons as nodes and document icons as leafs. Double-clicking over each folder will expand or collapse its content.

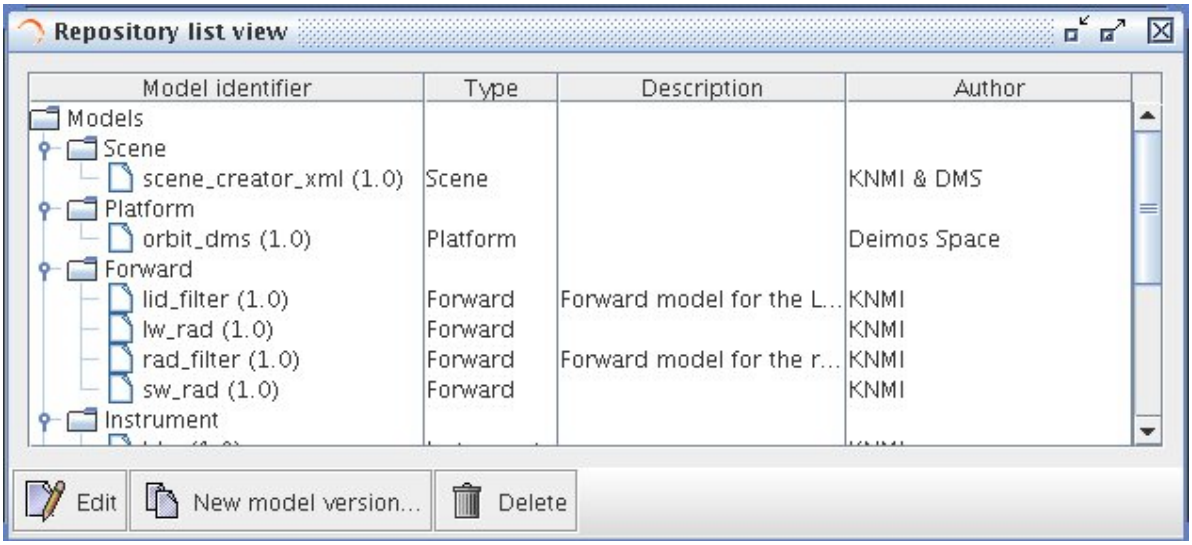


Figure 4-22: Model list view

© DEIMOS Space S.L., 2007

4.8.3. Adding models

Users can add a new model accessing this functionality from the main menu or alternatively clicking over the "New" button in the model manager.

This frame contains the components needed to introduce all data to define a new model in the system. These data (model attributes) are grouped with the following structure:

- General
- Configuration
- Input / Output

Each category is analysed in the following sub-sections.

4.8.3.1. General data

In this group (Figure 4-23) users must define general information about the model to create:

Attribute Name	Format	Purpose	Sample
Model ID	Medium string	Unique model identification.	"LIDAR"
Model version	Float	In a new model this field will be filled with a default value. Users cannot edit this unless they use the "new version" functionality.	1.0
Description	Long string	Free writing area where to briefly describe the model.	"State-of-the-art" LIDAR instrument model
Author	Medium string	Text field where to write the author's name.	"DMS"
Type	Options: retrieval algorithms, instruments, platforms, forward models and scene creators	Model type or "family".	"Instruments"

© DEIMOS Space S.L., 2007

Attribute Name	Format	Purpose	Sample
Instruments compatibility	Options: ATLID, CPR, BBR, MSI	Every model is compatible with at least one of the EarthCARE instruments. For example: a model of a LIDAR should be compatible with the ATLID, a scene_creator or platform model must be compatible with all the instruments. Only compatible models can be selectable during simulation definition (see section 4.9.4 Simulation edition).	"ATLID"
Source code file	File	There is a text area to write the file name[5] and a button that will show a dialog to locate and choose the intended source code file.	"models/LIDAR/src/ lidar.f90"
Binary code file.	File	Same as above but referred to the corresponding binary[6] file.	"model/LIDAR/src/ lidar"

[5] Codes in general will have several routines. However, there will be always a "main". This function is interesting to pinpoint as it is a sort of manager for the rest of routines.

[6] The compiling process is an external procedure to ECSIM.

© DEIMOS Space S.L., 2007

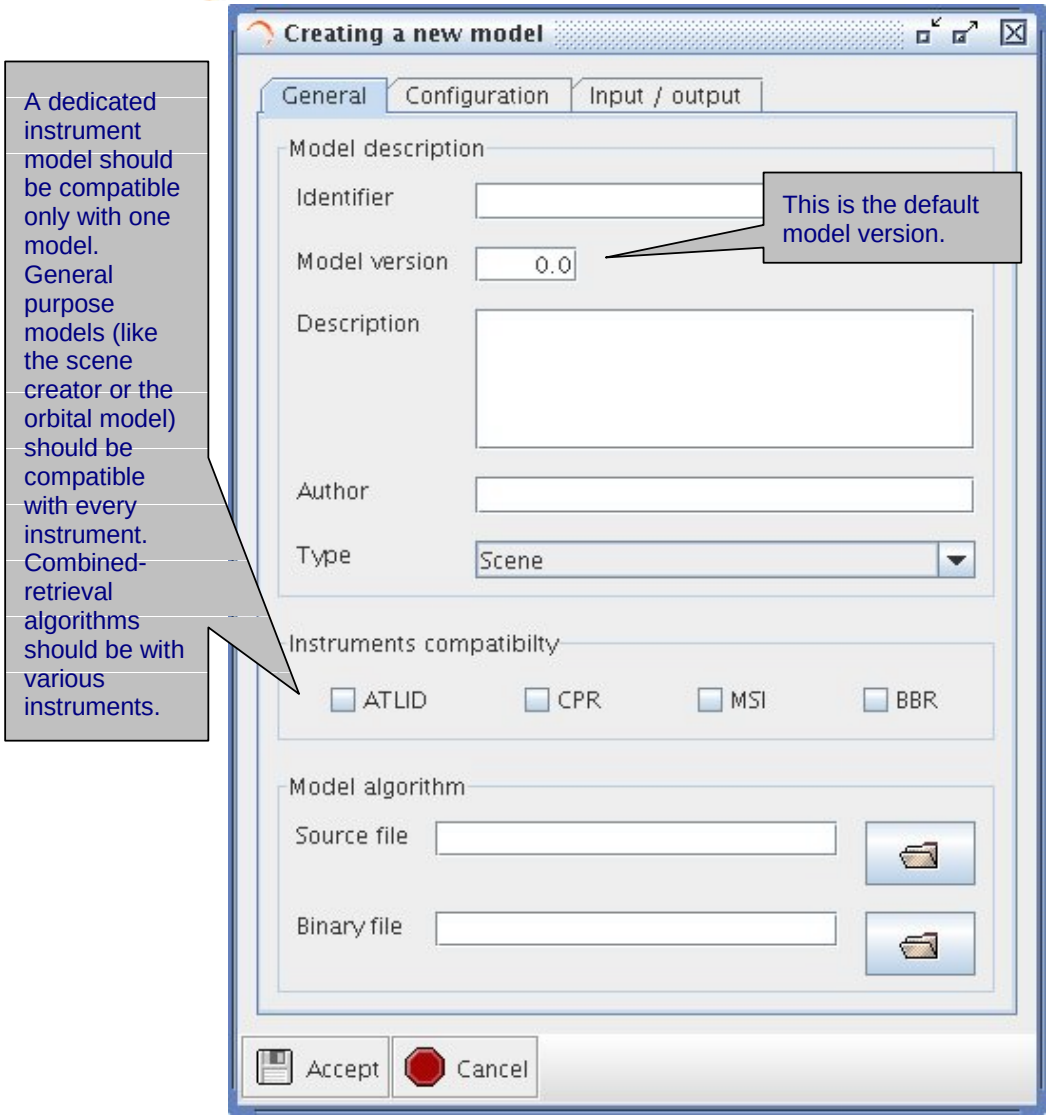


Figure 4-23: Model. General properties

4.8.3.2. Configuration

Selecting the "Configuration" tab (Figure 4-24), users can select the XML configuration file and its correspondent XSD schema file using file-browser dialogues. Text areas are also provided to preview the XML code.

© DEIMOS Space S.L., 2007

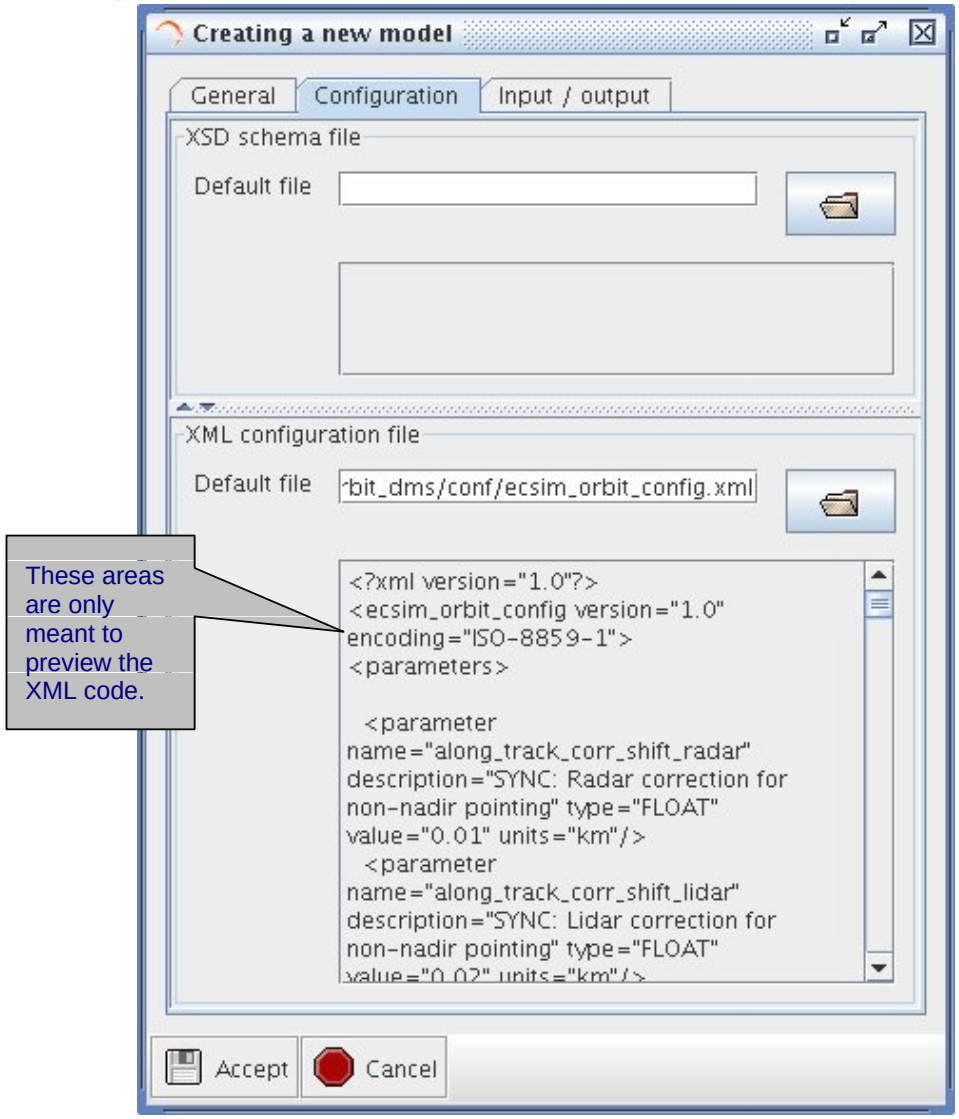


Figure 4-24: Model. Configuration

4.8.3.3. IO descriptors

The "Input" and "Output" tabs (Figure 4-25) from the Model properties window enable users to specify, respectively, the contents of the input files expected for the model, and the output files the model produces as output. Thus, users can select "Input" and "Output" tabs in order to define to the IO descriptors for this model. IO descriptors are intended to allow the model connection when defining simulations.

Each IO descriptor has an identifier that uniquely identifies the descriptor among the system. It may happen that an IO descriptor for a new model may exist already in the system, that is, this model uses the same type of files and file contents as another model. Therefore, a combo-box component is presented with the list of known IO descriptors in case the user desires to select an existing one.

© DEIMOS Space S.L., 2007

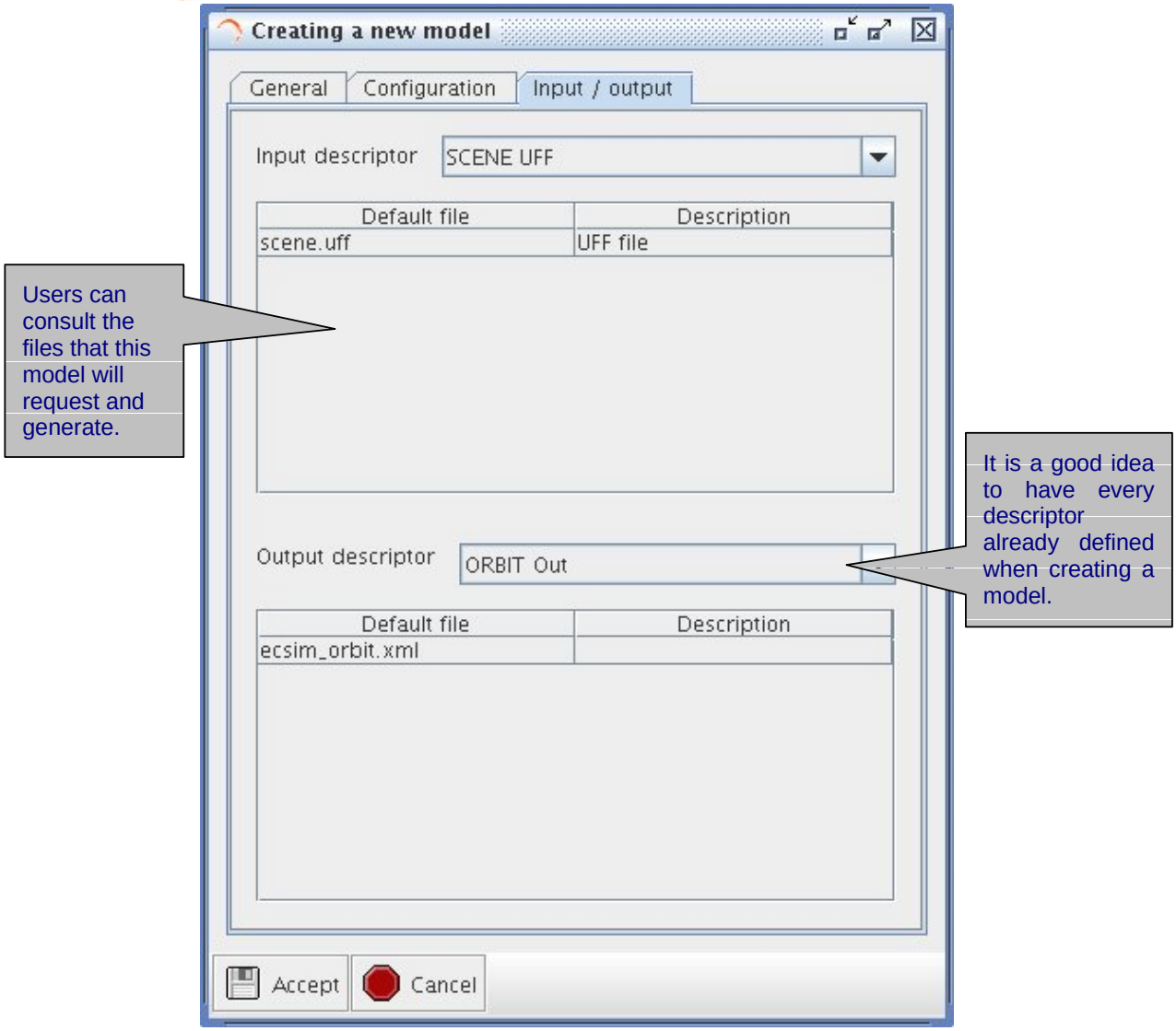


Figure 4-25: Model. Input/Output specification

4.8.4. Model upgrade - New version

A new version of a model represents an upgrade of the implementation of a given model. This means that users can alter the binary code of the model but not the configuration, input and output files originally involved in the model definition.

Users can create a new model version selecting the correspondent action from the context-menu of the repository view or alternatively clicking over the "new version" button in the model list.

The system will automatically perform a "minor version upgrade" (for example, from 1.0 to 1.1).

Users now can edit more attributes than with the "model edition" feature. Changeable attributes are the following:

- Model version.

© DEIMOS Space S.L., 2007

- Description.
- Author.
- Instruments compatibility
- Source code
- Binary code

4.8.5. Model edition

Editing a model from the repository view or from the models list, ECSIM will present the same window as in the previous chapter with all known data already filled. The window shall present fields in write-mode only the data that is susceptible to be modified (only description and author). If users want to change more attributes of a certain model, other operations must be used ("model creation" or "new model version").

This frame is intended to let users modify data of a certain model. Once they have finished with the edition, they can accept or cancel the changes made with the buttons at the bottom-side toolbar.

4.8.6. Model deletion

Users can select a certain model and choose the option to delete it. Once users confirm the operation the model is erased from the repository and the file system. Note that also every simulation, session (and its results) that uses this model will also be erased from the system for consistency purposes.

4.9. Simulations

As defined in section 1.3, a simulation is understood as a list of models (or even a model alone) that is run sequentially and produces observable results.

Users can access to the list of simulation known by the system in the repository view of the side bar or via the "Simulations" menu (Figure 4-26) from the main menu.

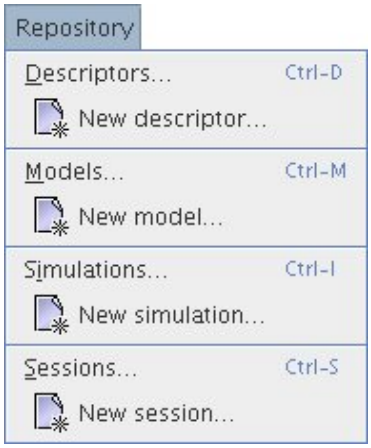


Figure 4-26: Simulations menu

Figure 4-27: Simulation. Pop-up menu

Operations involving simulations include the following:

- List – present the list of existing simulations;

© DEIMOS Space S.L., 2007

- Creation – capability to create a new simulation into the system;
- Edition – edit an existing simulation to enter changes;
- Deletion – delete a simulation from the system;
- Execution – Starts a new session definition including this simulation.

4.9.1. Simulation list

Users can access to the simulation list selecting the "List simulations" option from the "Simulations" menu.

Figure 4-28 shows an example of the simulation list window that is presented upon selection. Below the table including the simulations existing in the system, there is a tool-bar with buttons to access to the different functions listed previously. Users can thus select a certain simulation and perform the operations shown in the toolbar.

Data attributes shown in the simulation list table are simulation ID, description, the name of the author that created the simulation and both limits (start and end) of the simulation stages.

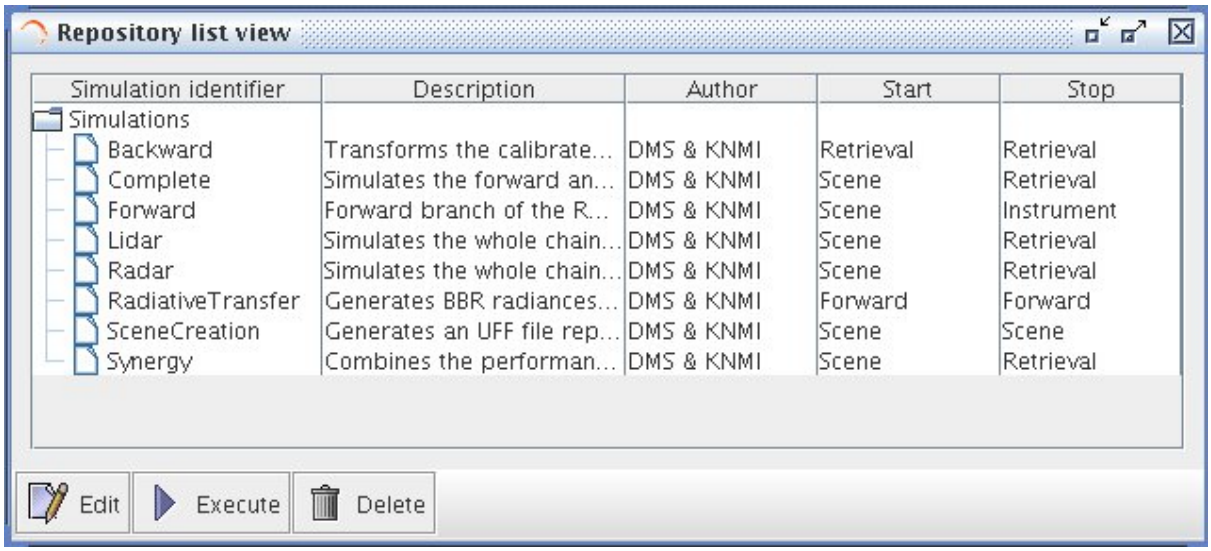


Figure 4-28: Simulations list view

4.9.2. Simulation creation

By selecting the "New simulation" option from the "Repository" menu the system will present a new frame (Figure 4-29) with components to introduce all necessary data to define a simulation.

© DEIMOS Space S.L., 2007

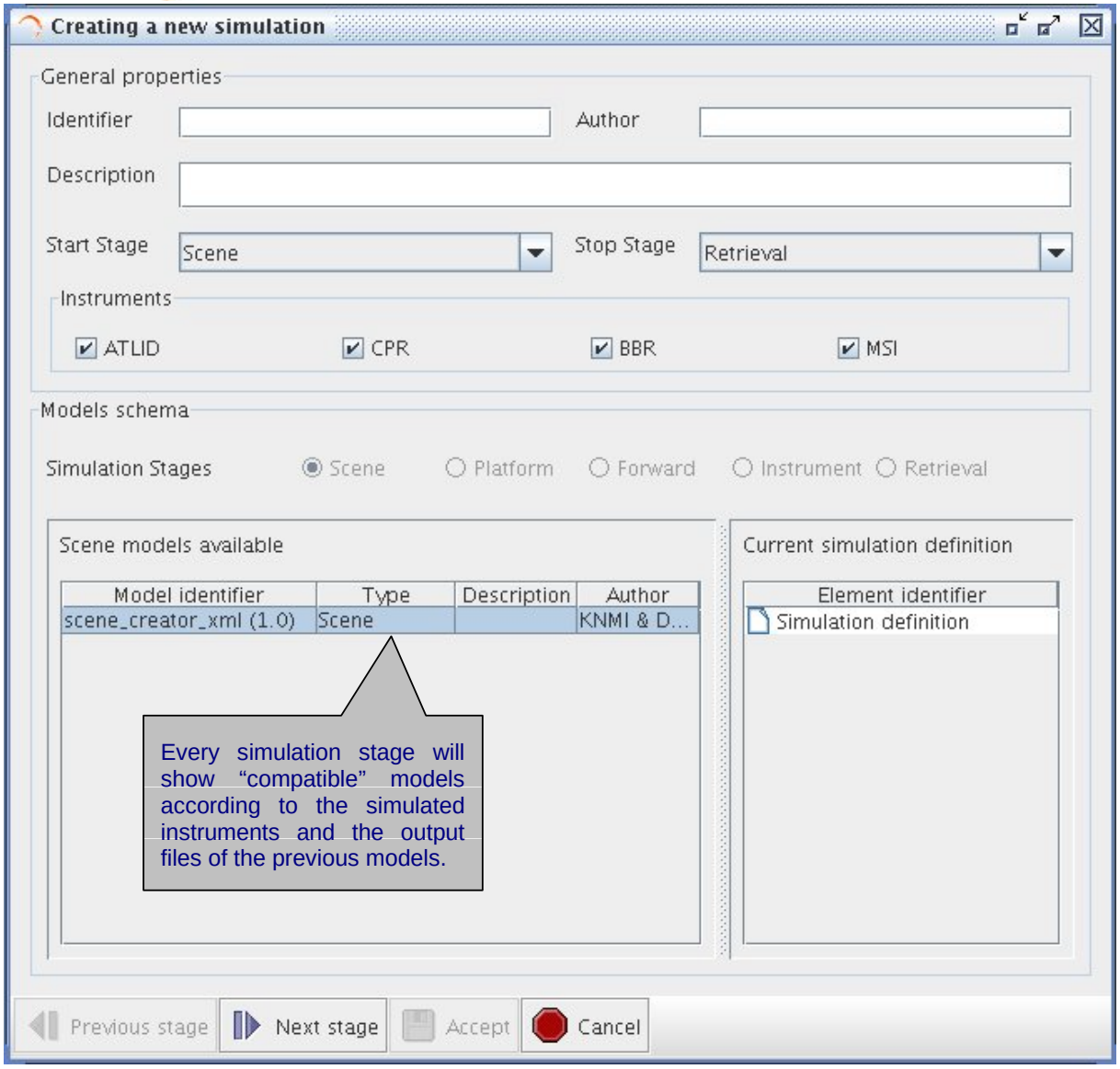


Figure 4-29: Simulation. Creation

The information needed to create a new simulation is organised in two areas separated in the window:

- **General properties**, containing general information about the simulation, and
- **Models schema** that allows defining the model constituents of the simulation. The series of models are listed in the "current simulation definition" tree.

Both spaces for the simulation creation are further detailed below.

4.9.2.1. General properties

The general data is constituted by the following attributes:

© DEIMOS Space S.L., 2007

Attribute name	Format	Purpose	Sample
Identifier	Short string	Identifies a simulation within the system.	"Lidar"
Author	Medium string	Denotes who created the simulation.	"DMS"
Description	Long string	Users can briefly describe a simulation in this text area.	"End-to-end simulation of a single instrument"
Starting and ending stages	Options: scene, platform, forward, instrument, retrieval.	Two combo-box components allow users to specify the first and last stage of this simulation. Stages coincide with model families that are scene, platform, forward, instrument and retrieval. Starting stage shall not be posterior to the ending stage and, reciprocally, ending stage cannot be anterior to the starting one. Updates shall be performed to the current simulation definition (sequence of models to be included) according with the changes of the limits.	"Scene" to "Retrieval"
Instruments	Options: ATLID, CPR, BBR, MSI	In order to ease the simulation definition, users can filter models by the instruments that want to include in the simulation. Thus only model compatible with selected instruments shall be selectable for current simulation.	"ATLID"

Note that selecting at this stage the instruments that will take part of the simulation, marks the "instrument-driven" character of the simulation creation process.

4.9.2.2. Models schema

Users can now form the sequence of models that comprise the simulation, the model schema. Users have three different panels:

- **Simulation stages**. The list of simulation stages (model families) that the model schema must fill in according with the limits defined before. The current stage is also indicated in this panel. If no stage is indicated then the simulation is completely defined.
- **Stage available models**. A list of available models from which the user selects one in order to start defining the simulation chain. Notice that the list of models that appears in the panel must conform to the models corresponding to the previous processing step derived from the simulation type and the instruments compatibility (this is done automatically by the system using the model definition and the IO descriptors).

  It is important to note that the compatibility of a certain model is given by the fact that, at least, one file of its input is provided (i.e. have the same default file name) by the output of the previous models (not necessarily the preceding one).

© DEIMOS Space S.L., 2007

$$\bigcup_{i=0}^{j-1} Output_i \cap Input_j \neq \phi$$

"$Output_i$" is the set of files generated by a model; "$Input_j$" is the set of files needed by a model.

- **Current simulation definition**. A tree showing the current structure of the models that will perform the simulation. Models are shown ordered following the sequence that will be used later in the session execution. Models are ordered by its family and, internally into a family, by the IO descriptors. For example: combined retrieval algorithms shall be executed after single-instrument retrievals.

Users can select an individual model, a range of models or a list of ranges (using the left mouse button in combination with the shift and control buttons) from the list of valid models.

Eventually, the "Previous stage" and "Next stage" buttons are used to navigate through the models (e.g., scenes, instruments, etc.) that define the simulation. Upon selecting the "next stage" action, selected models from the list of valid models will be included in the "current simulation definition". Pressing the "previous stage" button, the simulation will be positioned in the previous stage and remark the previous selection of models.

4.9.3. Simulation creation example: creating a forward simulation of the active instruments

This is an example that illustrates "step-by-step" the definition of the forward branch of a simulation.

4.9.3.1. Step 1: Entering the model list tab

The user has chosen in the "General" tab to indicate the forward branch of the simulation process (from scene to instruments) as appearing in Figure 4-30. The models tab panel will first show a list of available models to create a scene, as the first processing step of an End-to-End simulation consists of the scene generation.

© DEIMOS Space S.L., 2007

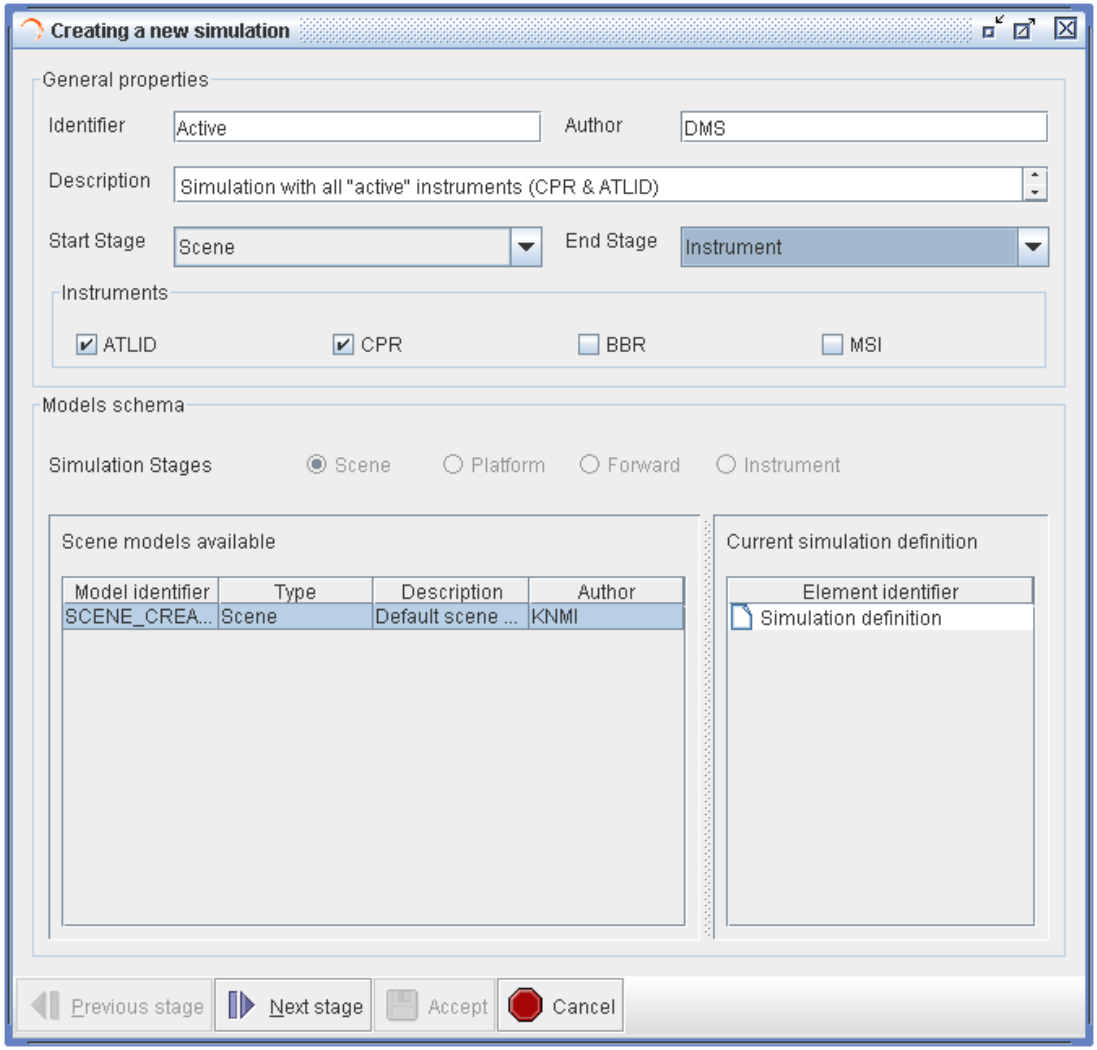


Figure 4-30: Defining a simulation - initial model window

4.9.3.2. Step 2: Selecting the first model of the simulation

The user can now select one of the scene generator models from the list (in the example there is only one available) by means of the "Next stage" button to add it to the simulation's structure. The result of this operation is illustrated in Figure 4-31.

© DEIMOS Space S.L., 2007

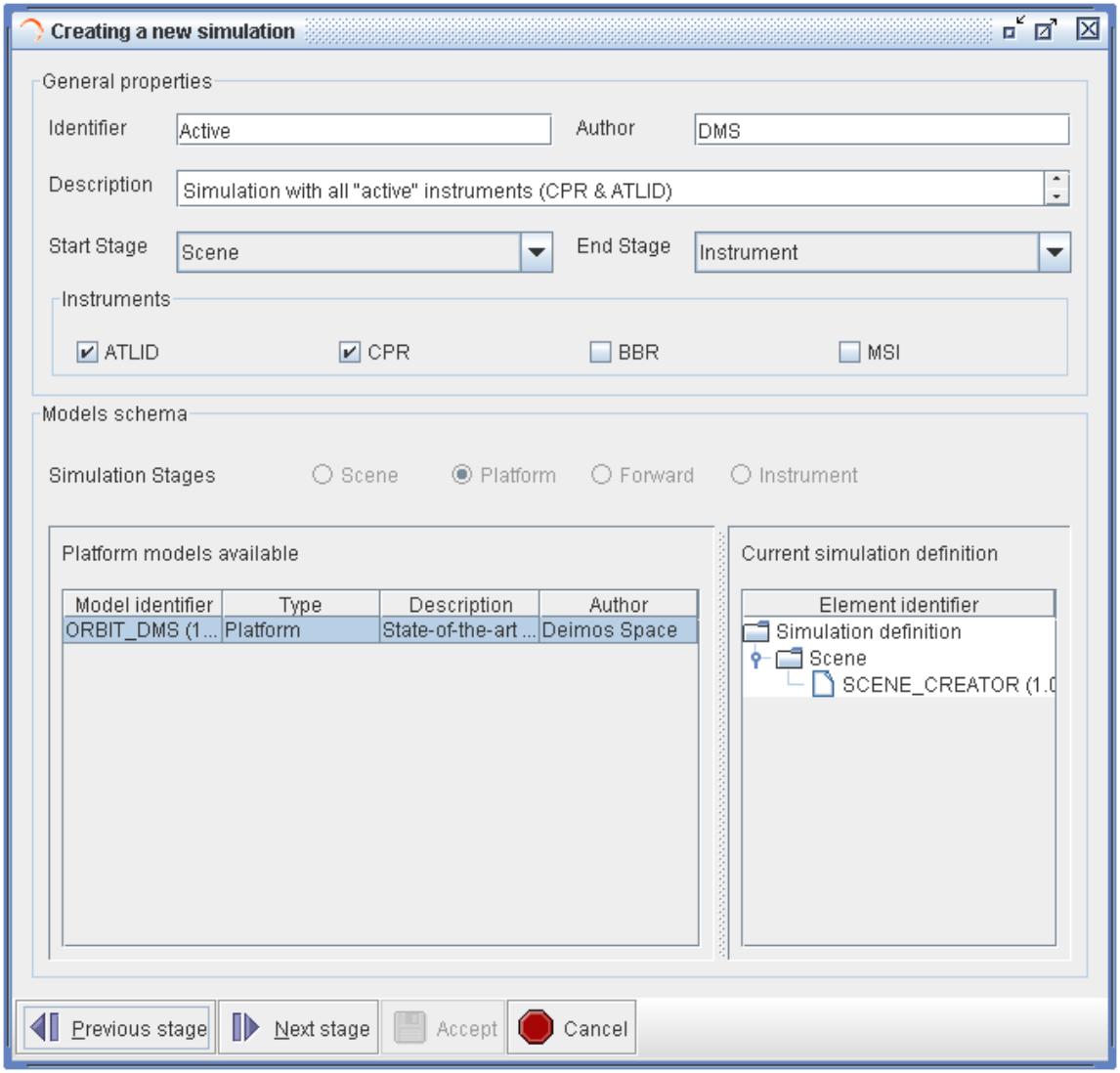


Figure 4-31: Defining a simulation - selection of the first model

4.9.3.3. Step 3: Selecting the remaining models of the simulation

The next step refers to the selection of the remaining models up to the completion of the simulation.

In the example, once the scene generator model has been chosen, the forward modelling must be defined. By pressing the "Next stage" button (Figure 4-32), the system will show the list of available forward models to join the simulation. It is important to mention that only those models that are compatible with the IO descriptors of the previous model are presented here. This means that during the simulation definition, the consistency of its constituents is granted.

Users can now select multiple models to add to the tree structure at the right side.

© DEIMOS Space S.L., 2007

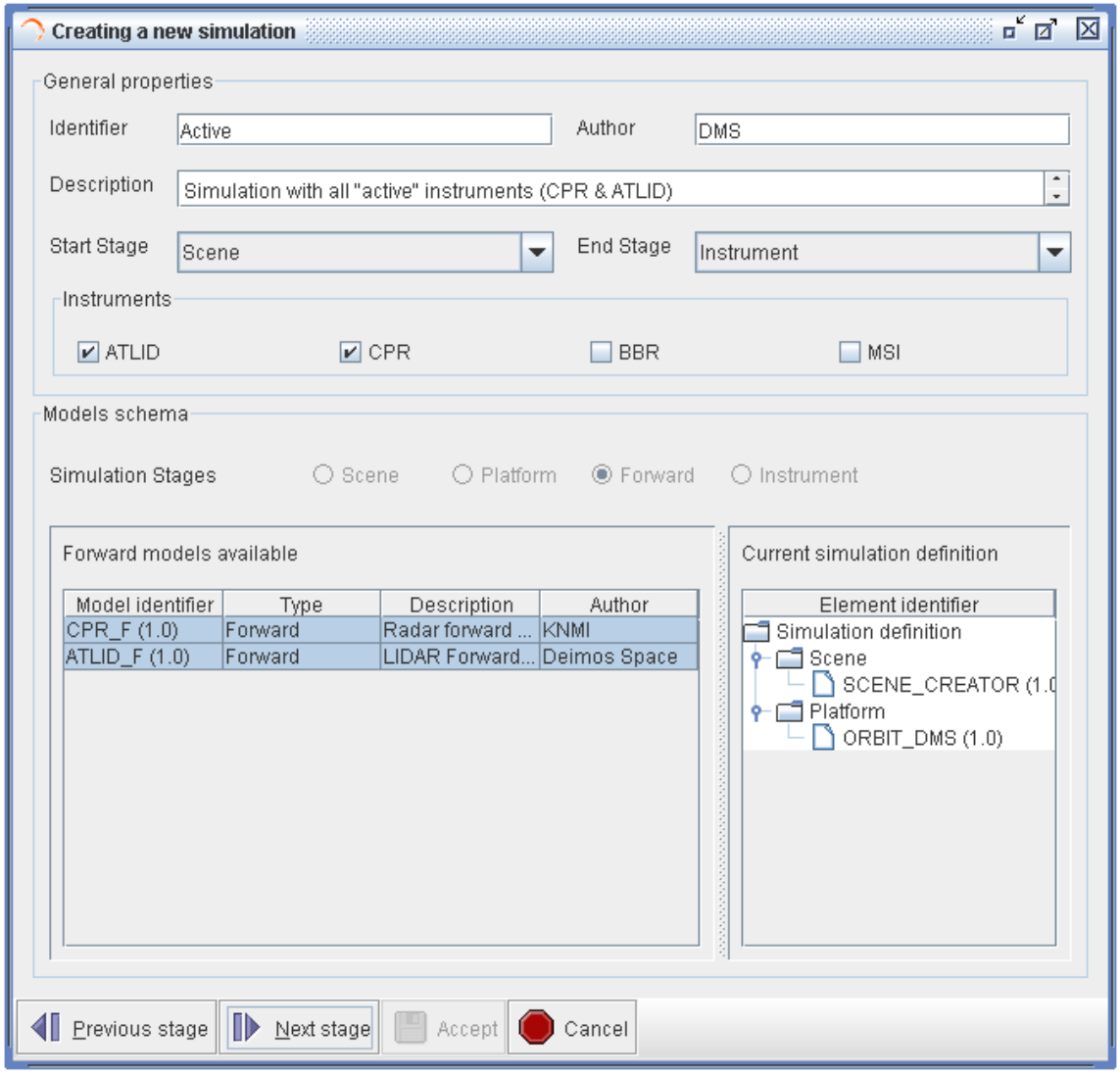


Figure 4-32: Defining a simulation – presentation of consistent models

Once all the desired forward models have been added to the structure, users can press the "Next stage" button and continue defining the models for, in this example, the platform, instruments and retrieval algorithms.

It is important again to recall that each time a model is selected, **the list of models presented in the next processing stage is limited to those that are compatible in terms of the IO descriptors**. Thus, the retrieval models shall be listed according to the selected instrumental model in the previous step. If the user would like to include retrieval models for another instrument, he/she must go back to the previous stage (or maybe selecting it from the tree structure) and select the corresponding instrument model.

© DEIMOS Space S.L., 2007

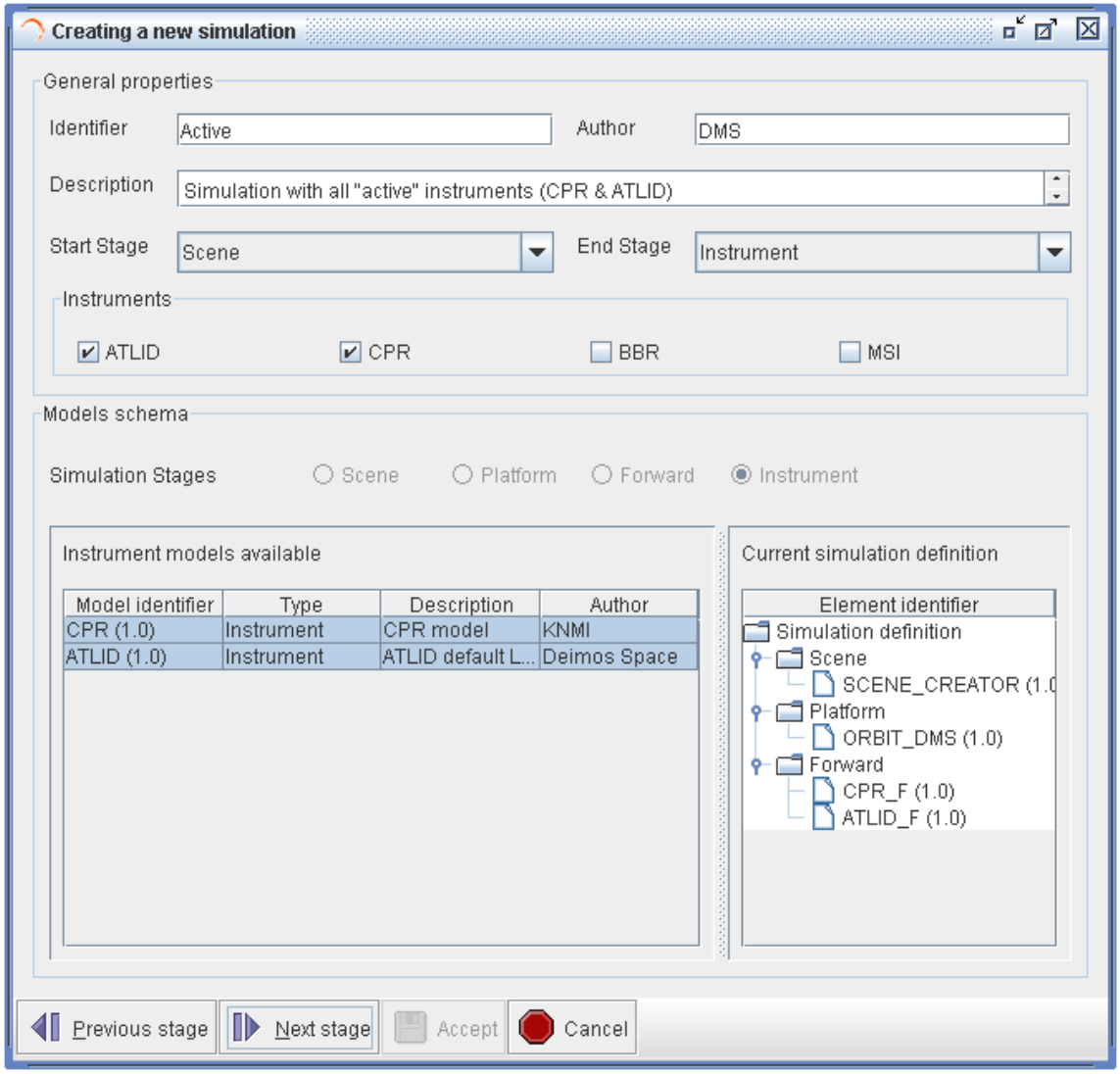


Figure 4-33: Defining a simulation – selecting models

4.9.3.4. Step 4: Finishing the simulation creation

Once all models for the simulation have been defined, the user may save[7] it into the database by clicking on the "Accept" button (Figure 4-34).

[7] Each type of simulation has to reach an ending point defined by its type. If with the available models this is not possible, then the simulation cannot be defined, and therefore it is not saved.

© DEIMOS Space S.L., 2007

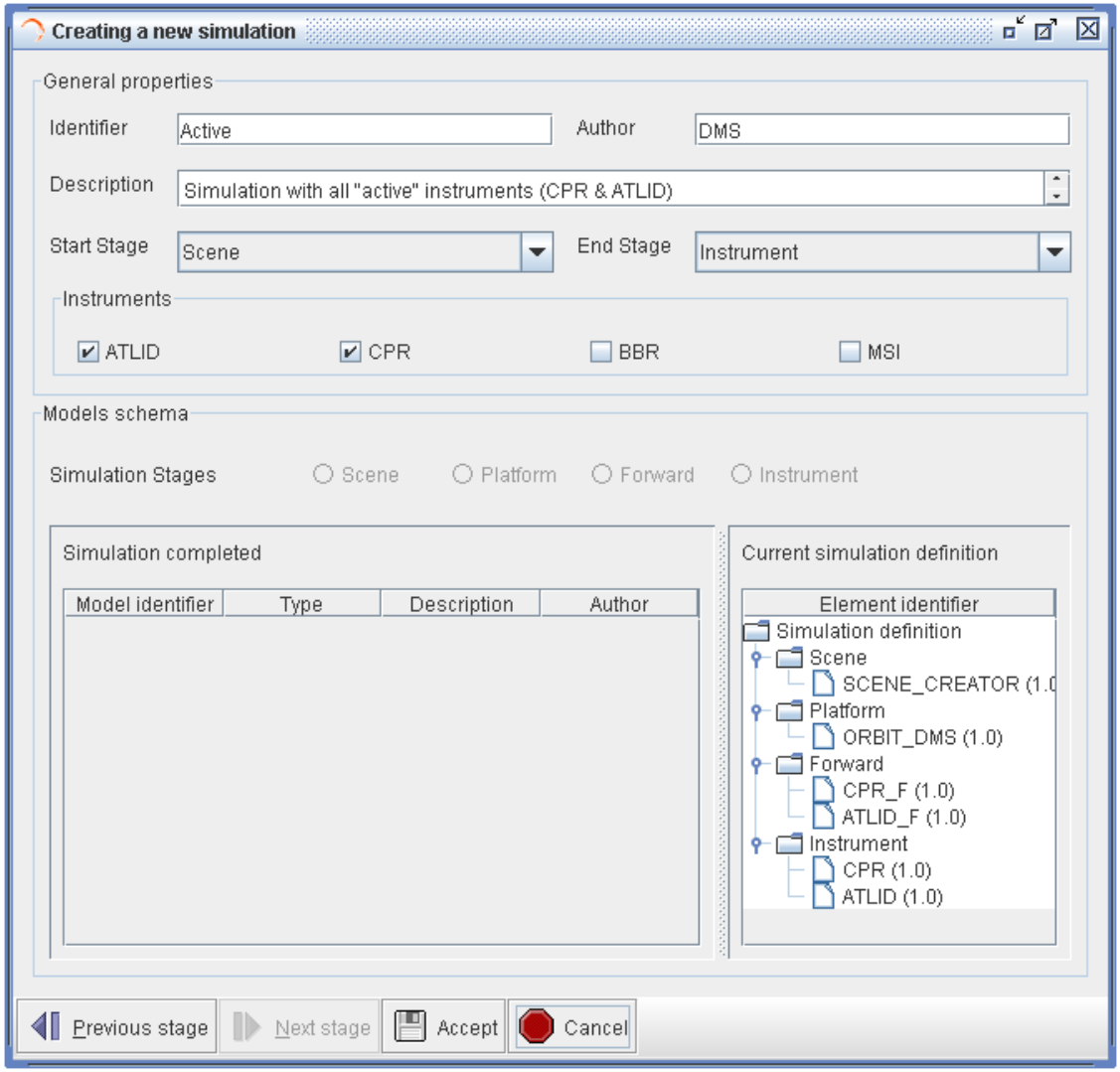


Figure 4-34: Defining a simulation – final window

4.9.4. Simulation edition

The edition of a simulation is accessed by either clicking on the "Edit" option from the main menu bar or selecting it from the context-menu at the repository view, or selecting the option from the actions tools bar of the simulation list frame.

The MMI will show the same frame as for "simulation creation" with all data corresponding to selected simulation already filled in. In this occasion only the description can be changed.

Once the user has finished with the simulation edition, the changes can be either accepted or cancelled making use of the corresponding buttons at the bottom-side toolbar.

4.9.5. Simulation deletion

Users can select a certain simulation and choose the option to delete it. The user needs to confirm the operation to completely delete the simulation from the repository. **Note that for consistency purposes,**

© DEIMOS Space S.L., 2007

the deletion of a simulation implies also the removal of every session (and its results) that uses this simulation from the system.

4.9.6. Simulation execution

The execution of a simulation is accessed by either selecting "Execute" from the context-menu at the repository view, or selecting the option from the actions tools bar of the simulation list frame.

This functionality is meant to rapidly create a session from a given simulation. Thus, the system will show a window like the example presented in Figure 4-35. This window corresponds to a session creation, but the selected simulation is already included in the simulations set and the description and author fields are filled with information extracted from the simulation. The "Run" button at the bottom of the window launches the execution of the session.

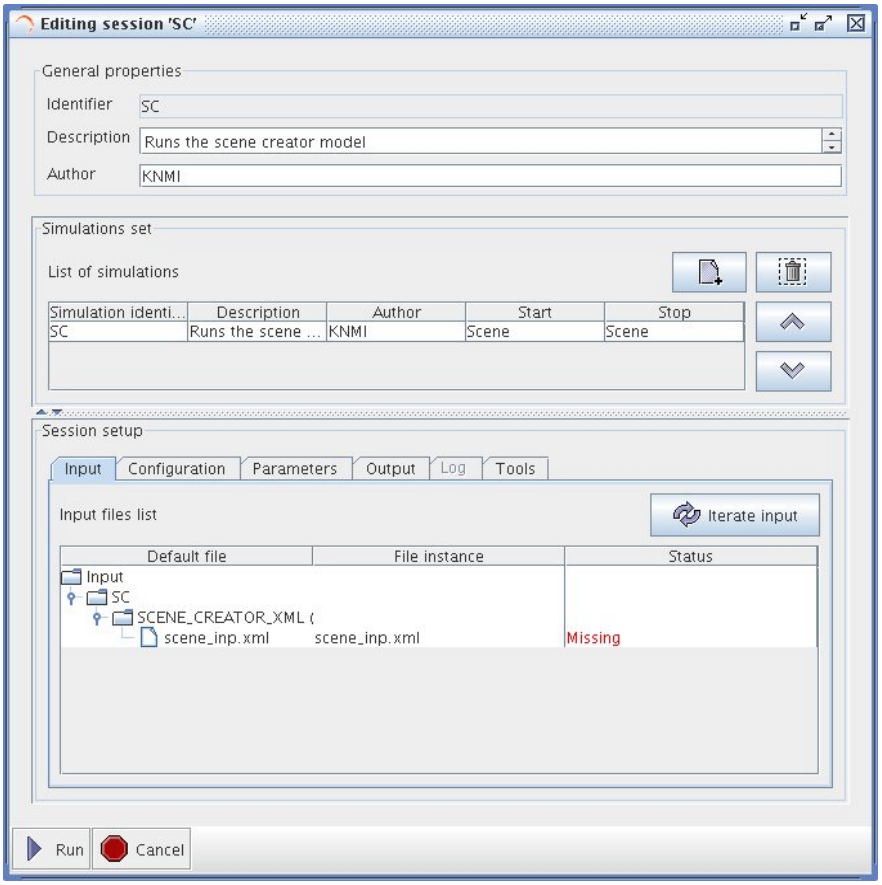


Figure 4-35: Simulation. Execution

4.10. Sessions

According to section 1.3, a session is defined as an execution of either one simulation, an ordered set of simulations, or an iterative execution of simulation/s with different parameter values.

Users can access to the list of sessions existing in the system in the repository view of the side bar or via the "Sessions" menu (Figure 4-36) from the main menu.

© DEIMOS Space S.L., 2007

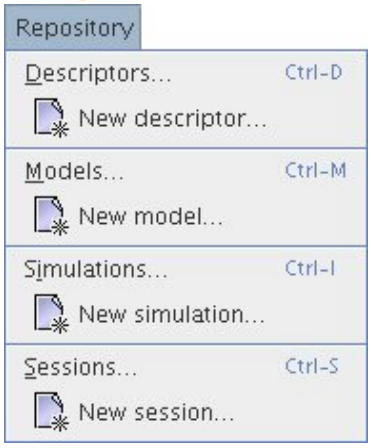


Figure 4-36: Session menu

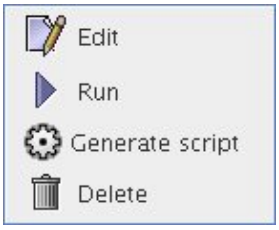


Figure 4-37: Session pop-up menu

Operations involving sessions include the following:

- *List* – present the list of existing sessions;
- *Creation* – capability to create a new session into the system;
- *Edition* – edit an existing session to enter changes;
- *Deletion* – delete a session from the system;
- *Run* – Starts a new session execution;
- *Script generation* – creates and stores a script describing the session.

4.10.1. Session list

Users can access to the session list selecting the "List" option from the "Sessions" menu.

Figure 4-38 shows an example of the session list window that is presented upon selection. Below the table including the sessions existing in the system, there is a tool-bar with buttons to access to the different functions listed previously. Users can thus select a certain session and perform the operations shown in the toolbar.

Data attributes shown in the session list table are session ID, description and the name of the author that created the session.

© DEIMOS Space S.L., 2007

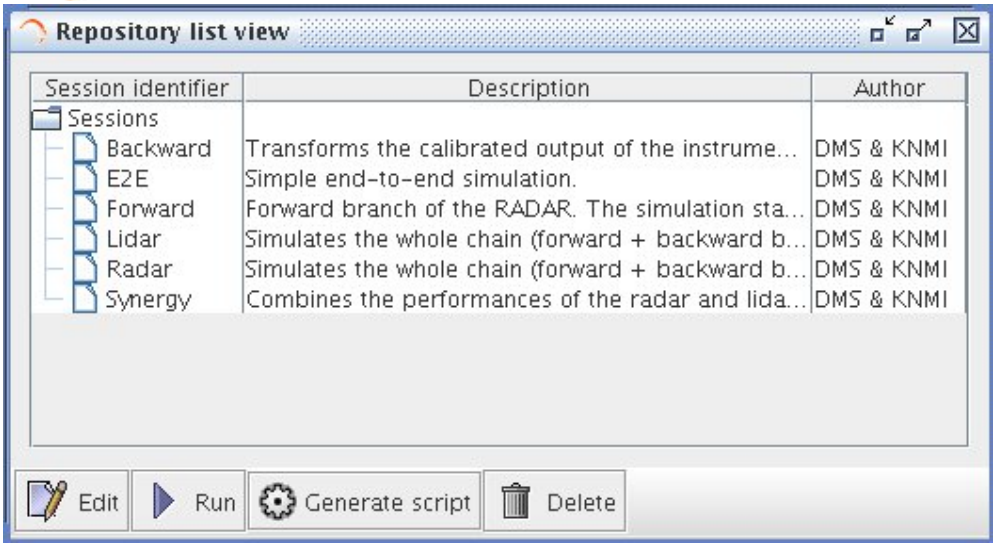


Figure 4-38: Session list view

4.10.2. Session creation

Users can create a new session definition by clicking on the corresponding action from the Session menu.

4.10.2.1. Import a session folder

The first thing the system will ask the user is the possibility to import an existing session folder, as shown in Figure 4-39.

Session creation
You can create a new session importing settings from an existing folder.
Do you want to import?
Import
Create new

Figure 4-39: Session import

Once the user selects the "import" option, the created session will have as identifier the same name of the folder and files within can be used as input for the session.

4.10.2.2. General properties

Figure 4-40 shows a blank session creation window. First thing the user must fill shall be the following general properties:

Attribute name	Format	Purpose	Sample
Identifier	Medium string. No blank spaces allowed.	Uniquely identifies this session definition into the system.	"E2E"

© DEIMOS Space S.L., 2007

Attribute name	Format	Purpose	Sample
Description	Long string	Brief remarks about the goals and characteristics of the simulation	"This is a full end-to-end simulation for the EarthCARE mission"
Author	Medium string	Name of person or group responsible of the session definition	"DMS"

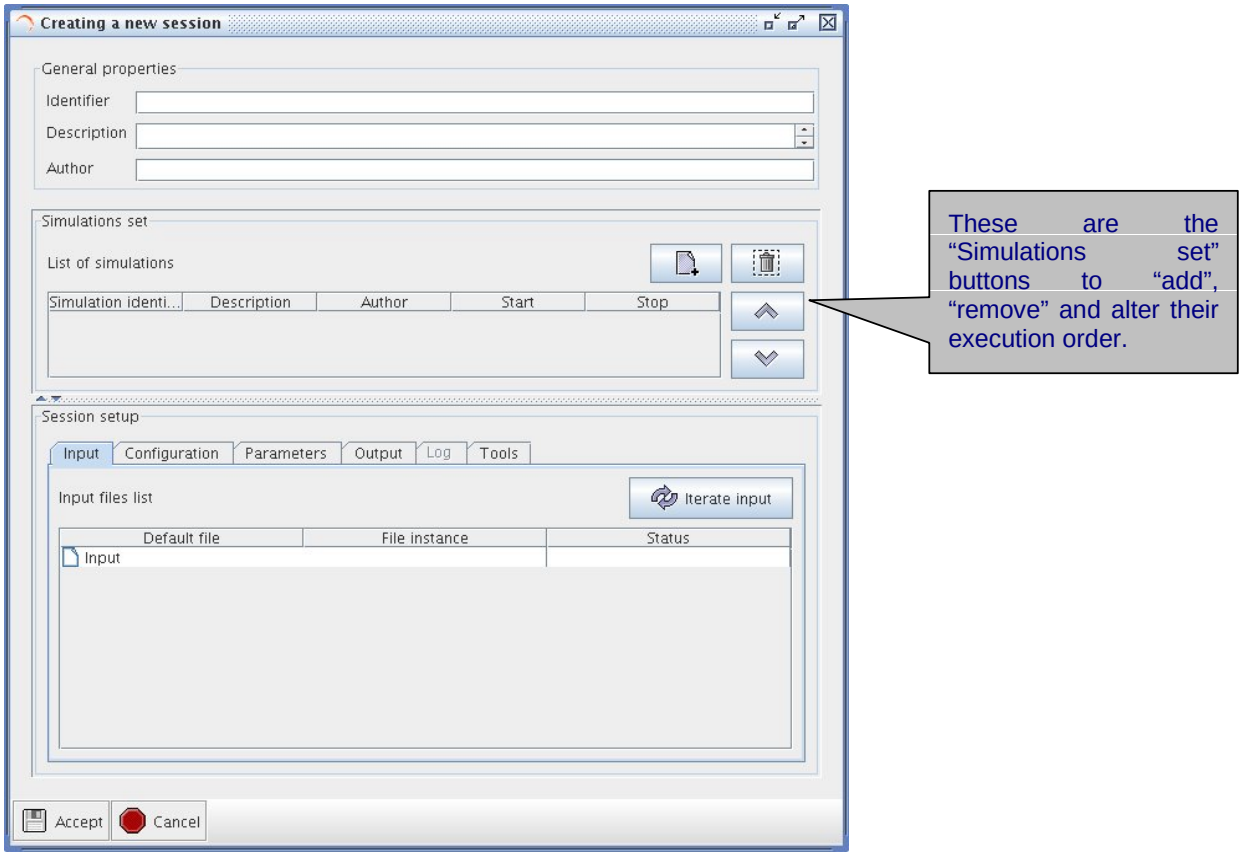


Figure 4-40: Session creation – General properties

4.10.2.3. Altering the simulation set

First, simulations to be executed need to be included by selecting them from a list of available simulations. Users just need to click on the desired row at the simulation list (Figure 4-41) and press the "Ok" button.

© DEIMOS Space S.L., 2007

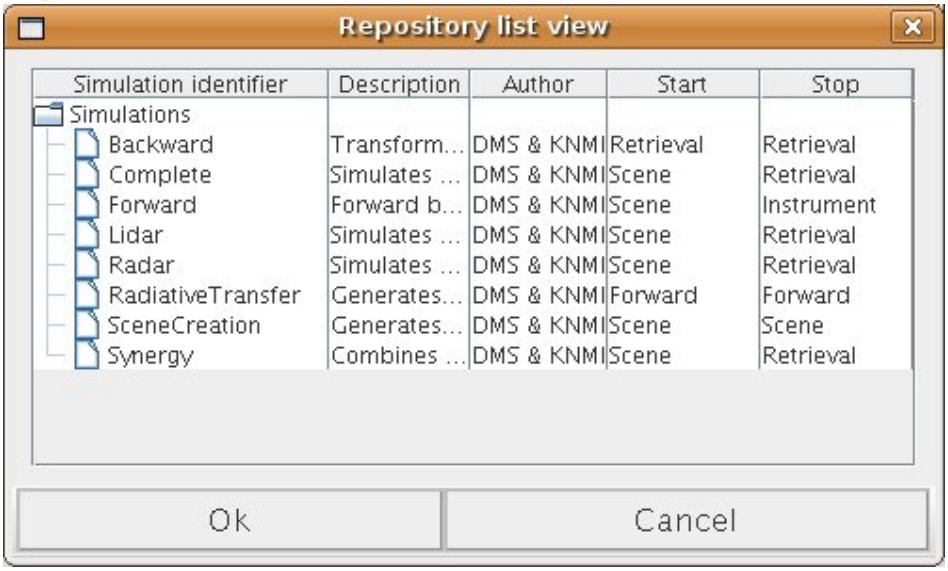


Figure 4-41: Session – Simulations specification

Users can also remove a desired simulation from the session's simulations set by just selecting it and clicking the "remove" button.

It is also possible to alter the simulations execution order. Users can move up and down a selected simulation from the list.

4.10.2.4. Provision of input data

Selecting the first tab of the session setup, (Figure 4-42) the system will ask for the location of the input file list needed to start the simulation.

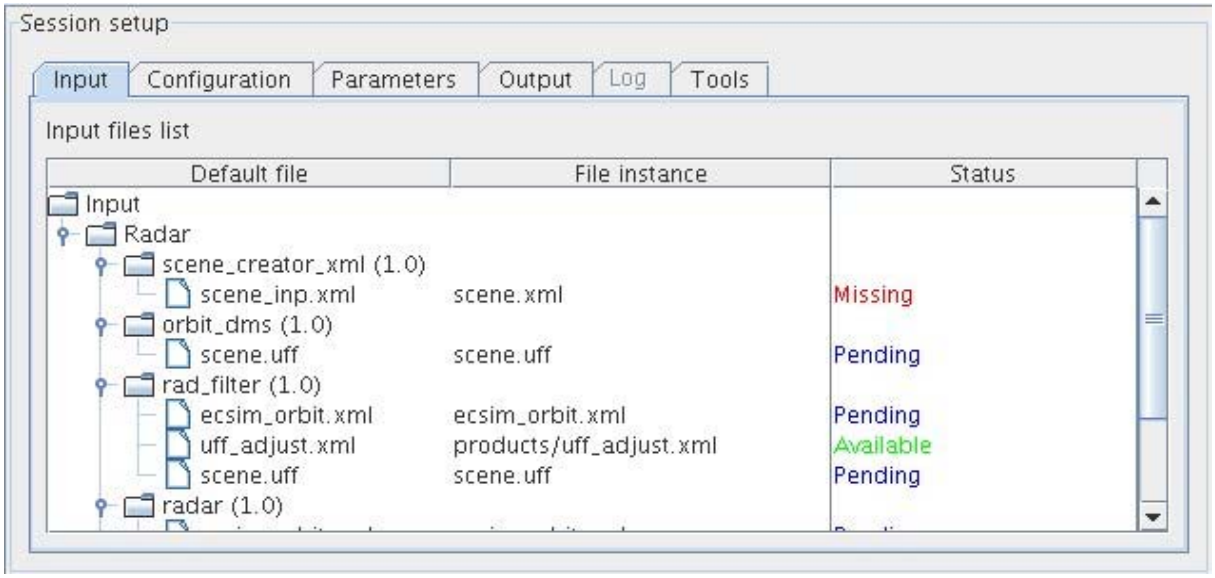


Figure 4-42: Session – Inputs definition

Double-clicking on the "file instance" column a file browser dialog will pop up. The user can now navigate through file system directories to specify the desired file's location. Note that at the time of

© DEIMOS Space S.L., 2007

defining the simulations, input files are defined from a general point of view (e.g., a file with ground altitude). It is in this step where a user can physically locate each file with the required data.

The "Status" column will show one of three different options for each file:

- Available (green) – the file instance is present, so the file is ready to be used for the model executions;
- Pending (blue) – the file is not present but it will be generated for the model executions before needed;
- Missing (red) – the file is not present and is not scheduled to be generated before needed. Edit the file instance and change it so that it appears as an existing file.

File instance locations can be specified using absolute paths or $ECSIM_HOME relative paths.

4.10.2.5. Provision of configuration files

As it is done at the "input data" step, the simulation needs to be provided with the location of configuration files needed by all the models involved in the simulation (Figure 4-43).

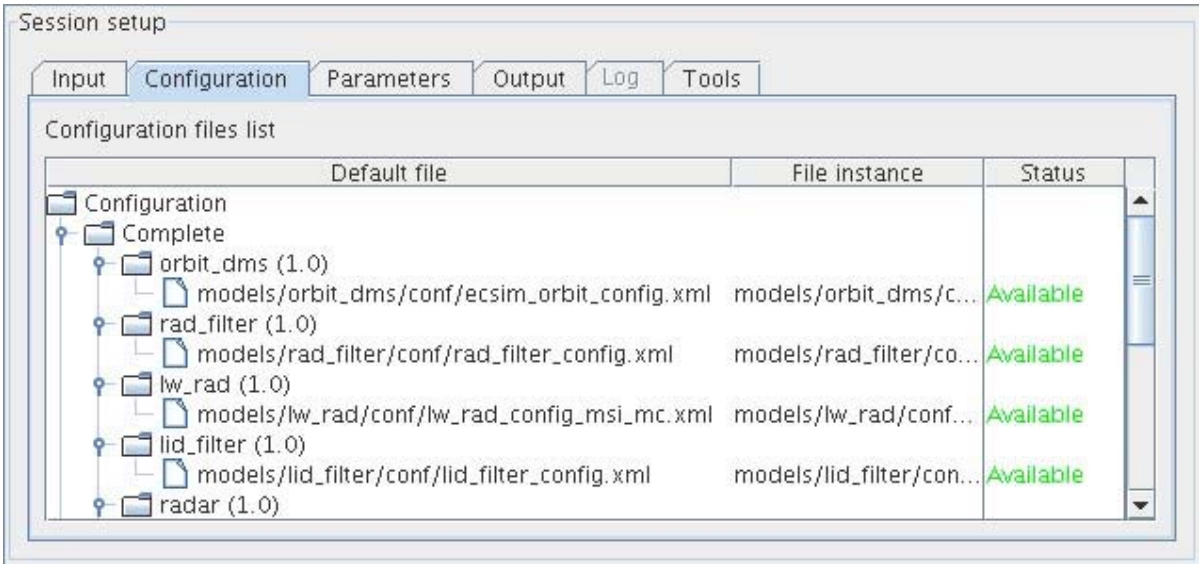


Figure 4-43: Session – Configuration files definition

This window will present the list of models present in the simulation and will ask for the location of each needed configuration file. At this stage the models cannot be changed nor deleted.

Double-clicking on a file row, the system will show a file browser to locate a specific configuration file. Providing an existing file will update the status to "Available".

4.10.2.6. Parameters configuration

This time, the user is able to alter the contents of the configuration files to change the behaviour of the model.

Assumed that the model has been correctly integrated into the system and a valid configuration file is reported, this tab shown in Figure 4-44 will present the list of model parameter (and values) grouped by model and simulation (sorted in execution order).

© DEIMOS Space S.L., 2007

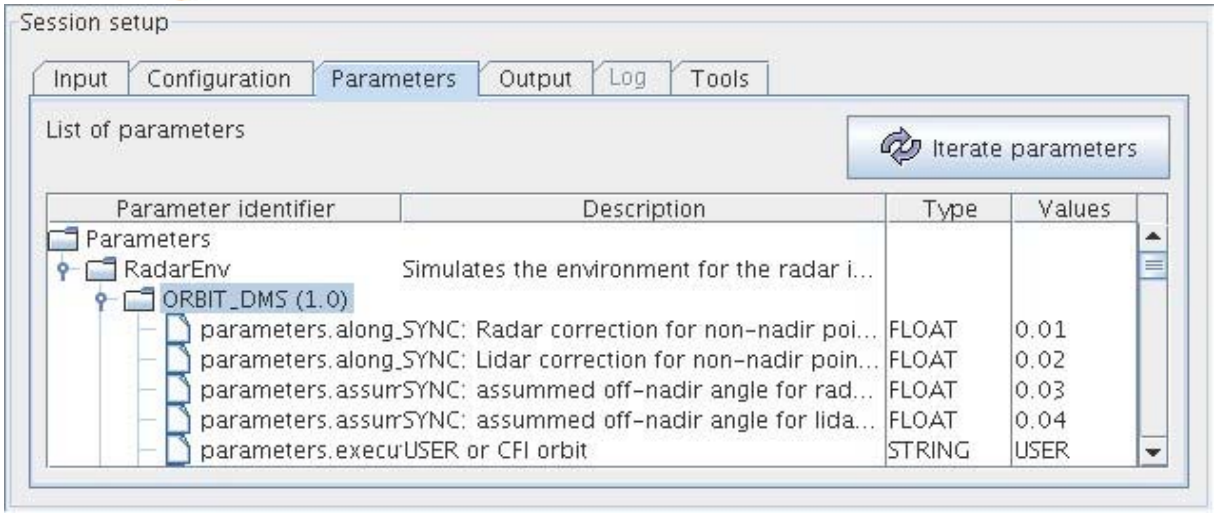


Figure 4-44: Session – Parameters definition

Users can consult the following list of attributes:

Attribute name	Format	Purpose	Sample
Parameter identifier	Long string	Complete name of the parameter. A parameter identifier is formed by its path into the file structure (dot separated) and its parameter name as described in [RD ICD].	"parameters. execution_mode"
Description	Long string	Brief description of the parameter purposes and values	"USER of CFI orbit"
Type	Options: INTEGER, FLOAT, STRING, BOOLEAN, FILE and FOLDER	Parameter values type. Used to present different editor when editing the parameter value	"STRING"
Values	(different types)	Parameter value. An unique value for each parameter is needed	"CFI"

Only the "values" column is editable to the user, the others just present useful information describing each parameter.

Note that changing the value of these variables will not affect to the "template" configuration files specified in the "Configuration" tab. The variables involved in a session definition are stored in database with the chosen values, meaning that the session will use them during the execution.

4.10.2.7. Specification of output files

Users can change the name and location of the output files that will be generated by execution of models. Selecting the "output" tab as shown in Figure 4-45, the system will show a list of output files grouped by models and simulations, and following the execution order.

© DEIMOS Space S.L., 2007

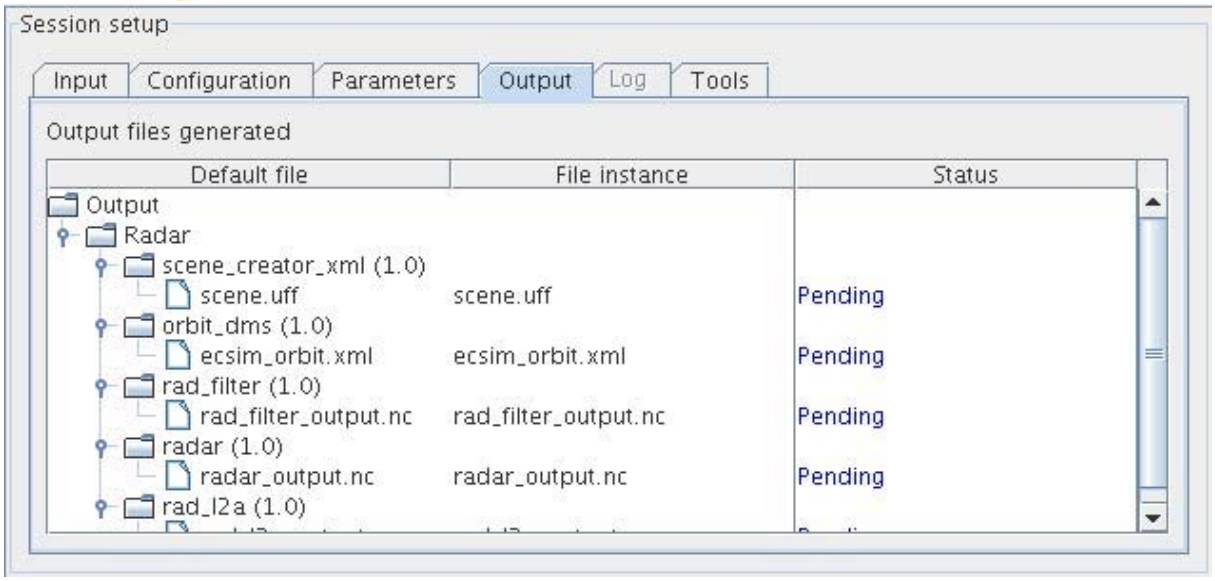


Figure 4-45: Session – Output definition

By default, these files will have a "Pending" status, meaning that they will be produced by the action of the models execution. Once a session has been executed and output files generated, file instance column will show the absolute path of the generated file and the status will be "Available".

4.10.2.8. Specification of final product tools

It is possible to add a list of product tools as post-processing operations, that is, a series of executables to be called upon the execution completion. For example, Figure 4-46 shows a list of tools set to graphically compare the reflectivity of an original scene and a reconstructed one as a RADAR instrument may sense it.

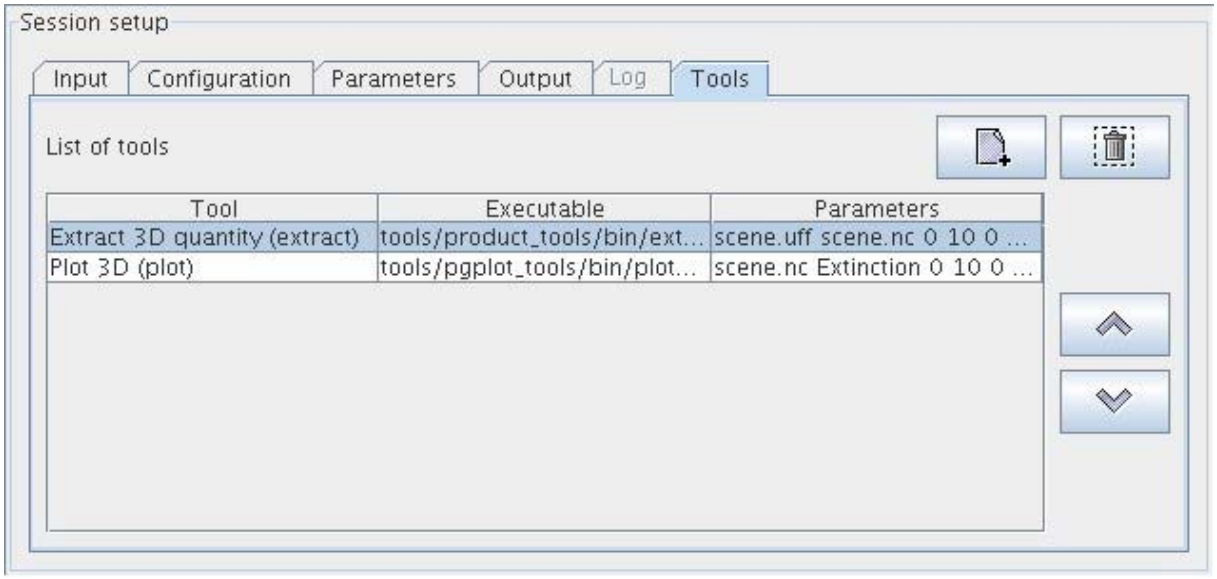


Figure 4-46: Session – Product tools specification

There are two ways to add tools to this list:

© DEIMOS Space S.L., 2007

- Selecting a file from the input, configuration or output files list. Users can right-click on a file marked as "available" or "pending" and a pop-up menu will appear. This menu will show a list of tools that can be applied to that certain file[8]. These tools can be executed instantly (if the file is already "available" or be scheduled to end of the execution process). Users can change the default parameters for the tool execution.

  When scheduling actions to certain files, ECSIM uses, instead of the actual file name and location, a reference to the file's foreseen location as an environment variable. These variables are named starting with the dollar symbol, then "IO", the simulation number and its identifier with no blanks, underscores or dots. For example, $IO0ecsimorbitxml denotes the orbital file to be generated in the proper folder by the execution process and $IO1radaroutput.nc, the NetCDF file generated by the radar model in the second simulation of a given session.

  It is also possible to use two other session-related variables: $SESSION_FOLDER to point the foreseen location of the session execution and $SIM_FOLDER_# (with "#" denoting the simulation number) pointing the foreseen location of a certain simulation in current session.

  In the same way, users can include references to the rest of ECSIM environment variables like $ECSIM_HOME.

- Clicking on the "add tool to session" button. Upon a selection of this action, users can choose one tool from the appearing list of defined tools. Users can change the default parameters for the tool execution.

Users can also select a certain tool and remove it from the list and, alternatively, change the order of execution of the tools with the arrow buttons besides.

4.10.2.9. Iterative sessions – iterating input/output files and parameter values

Users can assemble iterative sessions. This is a powerful feature that helps to run a large number of simulations by changing values of the parameters. Users can alter any parameter's value to fine-tune the behaviour of a model for a particular simulation run.

Selecting one or many files or attributes and pressing on the "iterate" button, the dialog shown in Figure 4-47 will come up. In this example we are going to iterate two float parameters from two different models.

[8] Tools defined in the system have an action associated to a file extension. See chapter 4.5.2.

© DEIMOS Space S.L., 2007

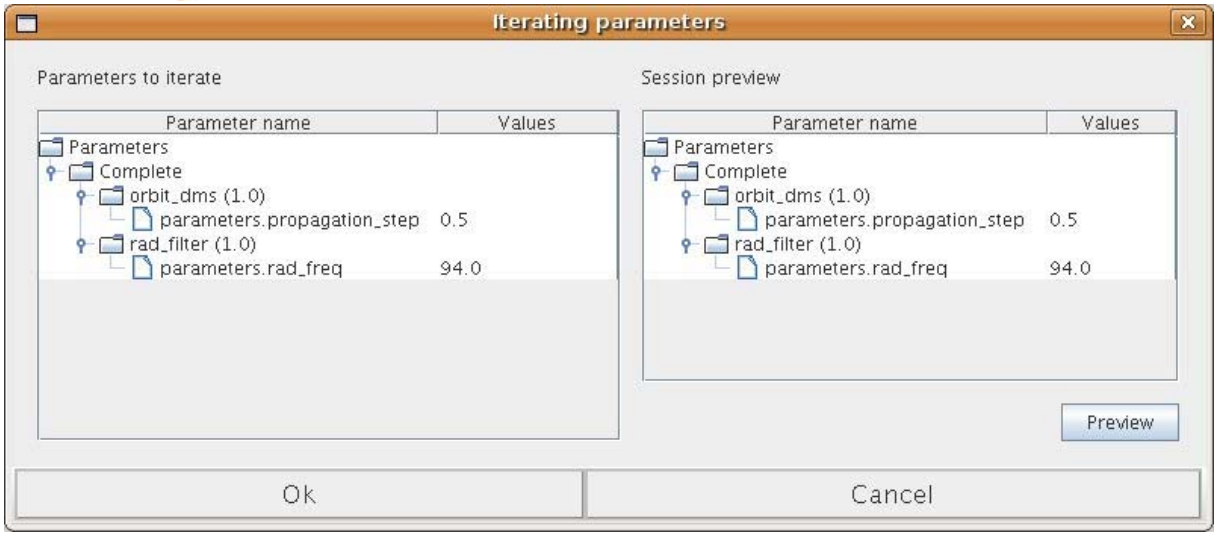


Figure 4-47: Session creation – Iterating parameters

This figure shows the initial state of the dialog. The list of selected parameters is in the left table and a preview of the models upon the combination of all the different parameter values on the right table.

Accessing to the "values" column of the left table, users can input a list of valid values separated only by blank spaces or commas, but not both.

Users can alternatively double-click on a parameter and the following "numeric sequence generator" dialog will show up:

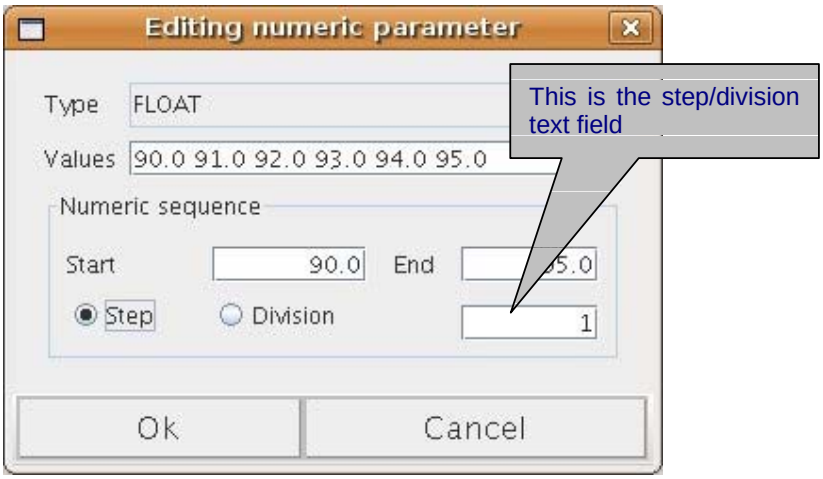


Figure 4-48: Session - Creation - Editing numeric sequences

This dialog lets the user define a numerical sequence of values (of the selected type: FLOAT or INTEGER) in three different ways:

- User input. Users can introduce their own values using the "Values" text field.
- Numeric sequence by step. Once defined the starting (x_1) and ending (x_n) values of the sequence, users can input the value of the step (s) in the step/division text field. Pressing the "generate" (marked with the ellipsis symbol) will create an arithmetical sequence following this rule:

© DEIMOS Space S.L., 2007

$$\{x_1, x_1 + s, x_1 + 2s, \mathrm{K}\}$$

Numeric values will never be greater than the upper limit. For example, a numeric sequence starting from 1 to 10 with a step of 5 will generate a series of (1, 6).

- Numeric sequence by division. Once defined the starting (x_1) and ending (x_n) values of the sequence, users can input the number of divisions (d) in the step/division text field. Pressing the "generate" (marked with the ellipsis symbol) will create an arithmetical sequence following this rule:

$$\{x_1, x_1 + s, x_1 + 2s, \mathrm{K}\}, s = \frac{(x_n - x_1)}{d}$$

Numeric values will never be equal or greater than the upper limit. For example, a numeric sequence starting from 0 to 10 with five divisions will generate a series of (0, 2, 4, 6 and 8).

Users can now accept or cancel the numerical sequence.

Once a valid set of values is input in both parameters, users can press the "preview" button to consult the simulations that will be added to the session.

Figure 4-49 shows all possible combinations of parameter values tested in the example. Once the user accepts the combination, pressing the "Ok" button will add the generated simulations to the simulation list. Note that parameters not involved in the iteration will remain fixed to a value but, they can be manually changed as seen in chapter 4.10.2.5.

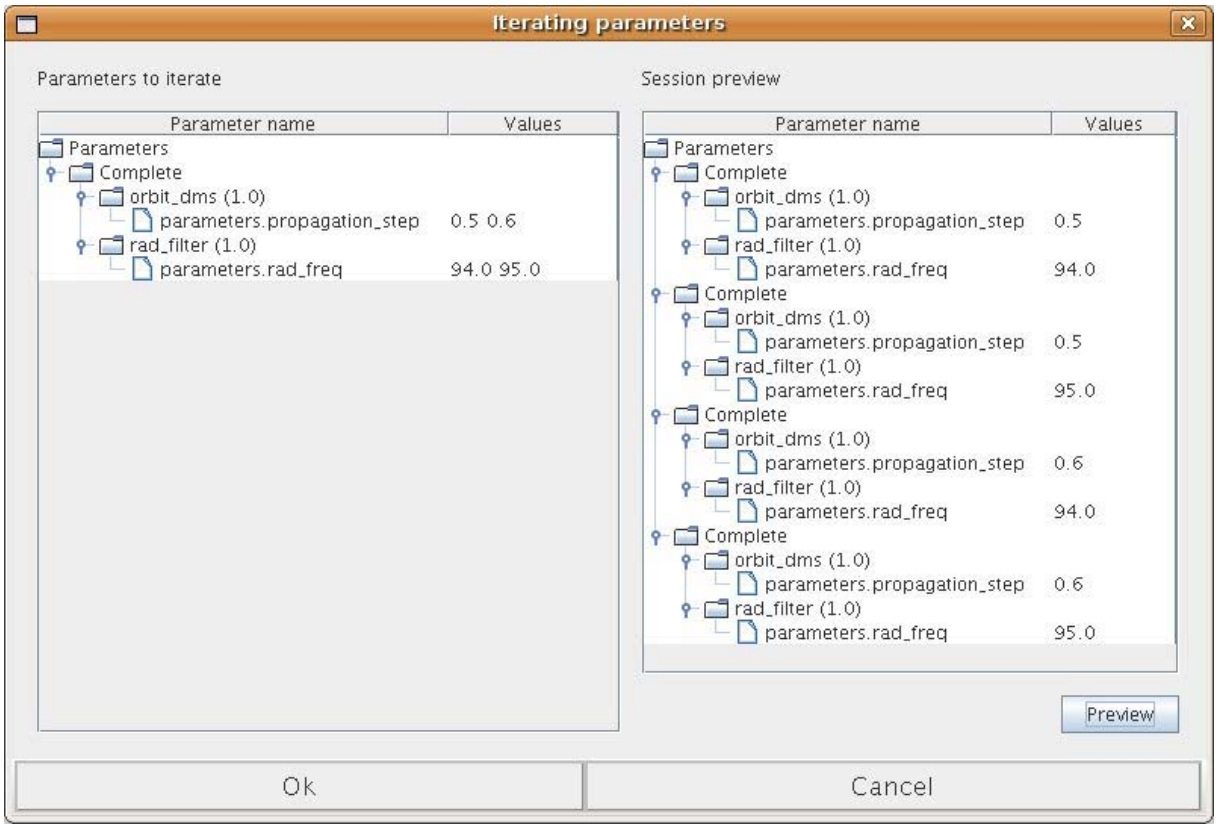


Figure 4-49: Session creation – Pre-viewing the parameters iteration

© DEIMOS Space S.L., 2007

Readers must know that the ECSIM system can filter redundant models out of an execution process. There is more information in chapter 4.10.5.

4.10.3. Session edition

It is possible to edit a given session and create a different one altering the information previously stored. Consequently, changes made to the session will not alter the previous but will create another.

When editing a session the system will show the same window as shown in the session creation. But this time, all the information concerning this session will fill every data field.

Once finished, the user can cancel the changes or execute this new session pressing the "Run" button.

4.10.4. Session deletion

Users can select a certain session and choose the option to delete it. Once the operation is confirmed the session is deleted from the repository and the file system. **It is to be noticed that for consistency reasons, the session deletion causes the removal of all results generated from that session from the system**.

4.10.5. Session execution – run

Once a session definition is ready, users can execute it.

Upon the activation of the "run" command, the system performs a series of checks to ensure the validity of the session:

- If there is any file with the "missing" status (that is, the system is unable to find in the given location), ECSIM will assume that this file will be in the right place when needed, so it leaves the responsibility of placing it in the correct place to the user or to process outside the system. A fatal error will be raised and the execution stopped if a model cannot locate the needed file.
- In some cases like executing iterative sessions or sequences of simulations with the same models, some of the models may be redundant (that is, they will generate the same output because they are run with the same input and configuration).

© DEIMOS Space S.L., 2007

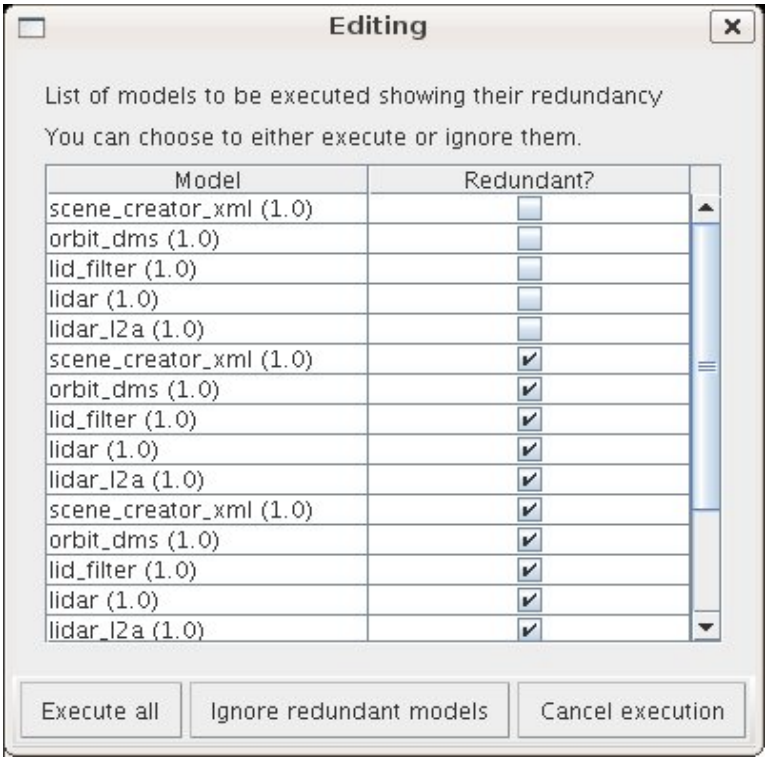


Figure 4-50: Session execution – Redundancy

Figure 4-50 shows the dialog that shall pop-up in case there is any redundancy. Users can choose to either execute or ignore all the models marked as redundant. Ignoring them saves a lot of processing time.

Once every validity check is fulfilled, a background process (Figure 4-51) will run the simulation and information concerning it will be given to the user in two different ways. One is the log window and another is the "simulation progress" window. The window presents a progress bar showing the percentage of simulation progress and a button to abort/close the execution.

© DEIMOS Space S.L., 2007

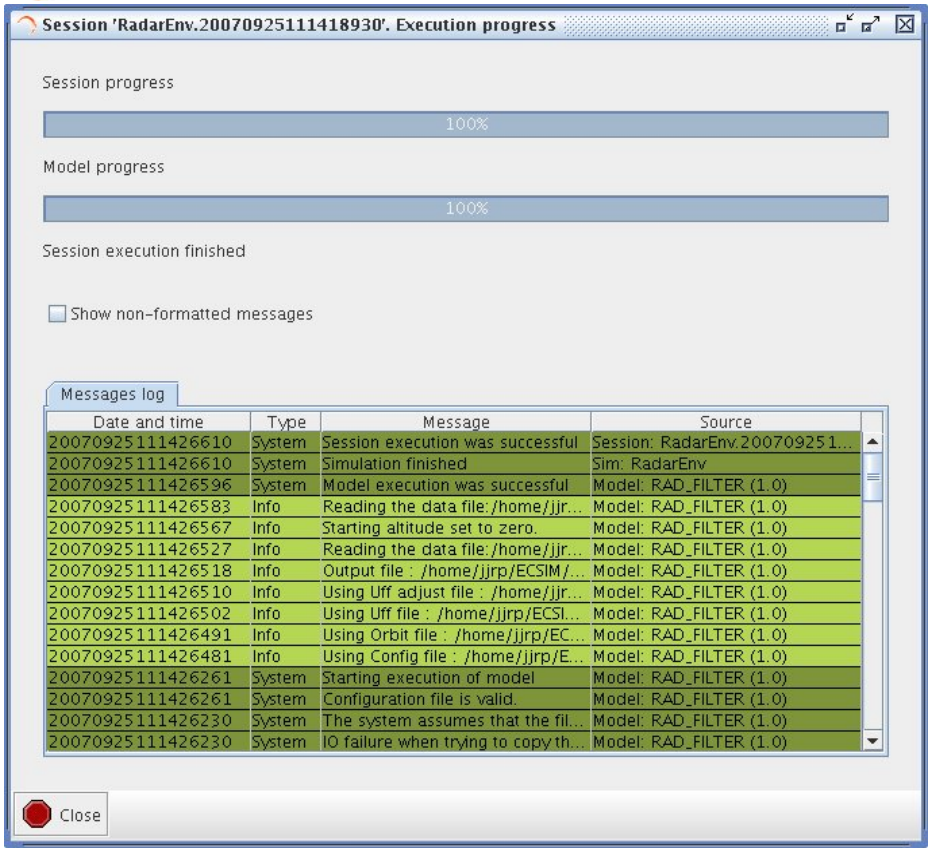

Session 'RadarEnv.20070925111418930'. Execution progress
Session progress
100%
Model progress
100%
Session execution finished
Show non-formatted messages
Messages log
Date and time | Type | Message | Source
20070925111426610 | System | Session execution was successful | Session: RadarEnv.200709251...
20070925111426610 | System | Simulation finished | Sim: RadarEnv
20070925111426596 | System | Model execution was successful | Model: RAD_FILTER (1.0)
20070925111426583 | Info | Reading the data file:/home/jjr... | Model: RAD_FILTER (1.0)
20070925111426567 | Info | Starting altitude set to zero. | Model: RAD_FILTER (1.0)
20070925111426527 | Info | Reading the data file:/home/jjr... | Model: RAD_FILTER (1.0)
20070925111426518 | Info | Output file : /home/jjrp/ECSIM/... | Model: RAD_FILTER (1.0)
20070925111426510 | Info | Using Uff adjust file : /home/jjr... | Model: RAD_FILTER (1.0)
20070925111426502 | Info | Using Uff file : /home/jjrp/ECSI... | Model: RAD_FILTER (1.0)
20070925111426491 | Info | Using Orbit file : /home/jjrp/EC... | Model: RAD_FILTER (1.0)
20070925111426481 | Info | Using Config file : /home/jjrp/E... | Model: RAD_FILTER (1.0)
20070925111426261 | System | Starting execution of model | Model: RAD_FILTER (1.0)
20070925111426261 | System | Configuration file is valid. | Model: RAD_FILTER (1.0)
20070925111426230 | System | The system assumes that the fil... | Model: RAD_FILTER (1.0)
20070925111426230 | System | IO failure when trying to copy th... | Model: RAD_FILTER (1.0)
Close

Figure 4-51: Session – Execution progress

The execution of a session raises a set of events. These events can be of one of these groups:

- *System information* – An event with some information to the user is generated by the platform. This is a harmless event, thus, the execution continues with no interruption. Coloured in dark green;
- *Information* – An event is raised by some model. Its message is intercepted and stored by the platform. This is a harmless event, thus, the execution continues with no interruption. Coloured in green;
- *Warning* – A model has detected a non-fatal error or situation that may cause a fatal error. This is a harmless event, thus, the execution continues with no interruption. Coloured in yellow;
- *Debug* – These events are raised when executing the session in "debug mode". This environment variable (defined in section 4.5.1) is optionally used by some models to show debugging information. Coloured in gray;
- *Error* – A fatal error has happened in the model execution and the model itself informs the platform about it, so the model has time to "graciously" close the execution. Another event that causes an error is that a model execution unexpectedly crashes, so this time the platform intercepts this error, informs the user and stops the execution. Coloured in red.

Pressing the "abort" button will make the system ask for confirmation. Once granted, the execution will be interrupted with an error event generated by the system. Later on, this session execution can be restarted or recovered from the last valid model executed.

© DEIMOS Space S.L., 2007

4.10.6. Session script generation

This functionality is provided to create and save a file script to enable the external execution of the session. This script file, called "<*session_name*>.sh", will be saved in the session folder as every needed input and configuration files. This script shall contain all the environment variables definitions and calls for models executions.

It is important to recall that while the execution will be mimicked, executing this script outside the ECSIM framework, the error handling and results storage capabilities will be lost.

4.11. Scenes

The ECSIM system can be divided in three logical parts: repository, scenes and executions.

The second of these parts is a set of different scenes, that is, ensembles of atmospheric data that defines the initial conditions of a simulation, serving as input files for the scene creator.

In this section, users will learn how to manage definitions of these scenes represented as XML files, that later will be used as UFF files taking part of simulations.

Users can access the scenes functionalities via the main menu bar (Figure 4-52) or via the scenes tab of the side bar (Figure 4-53). Right-clicking over a scene, the menu in Figure 4-54 shall pop-up.

Users can now access to all the following functionalities:

- *List* – present the list of existing scenes;
- *Creation* – capability to create a new scene into the system;
- *Edition* – edit an existing model to enter changes;
- *Deletion* – delete a model from the system;
- *Comparison* – functionality to show the differences between two scenes;
- *Merge* – combine two scenes to form a third.

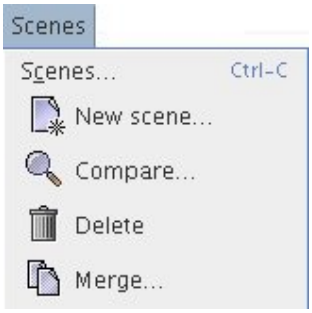


Figure 4-52: Scenes main menu

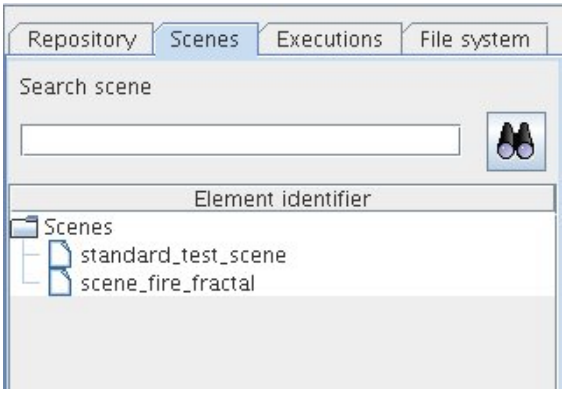


Figure 4-53: Scenes tab in the side bar

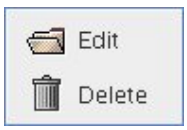


Figure 4-54: Scenes pop-up menu

4.11.1. List scenes

The list of scenes shown in the scenes tab of the side bar and in the figure below represents the valid XML files stored in the default scenes directory (see section 4.5.1).

© DEIMOS Space S.L., 2007

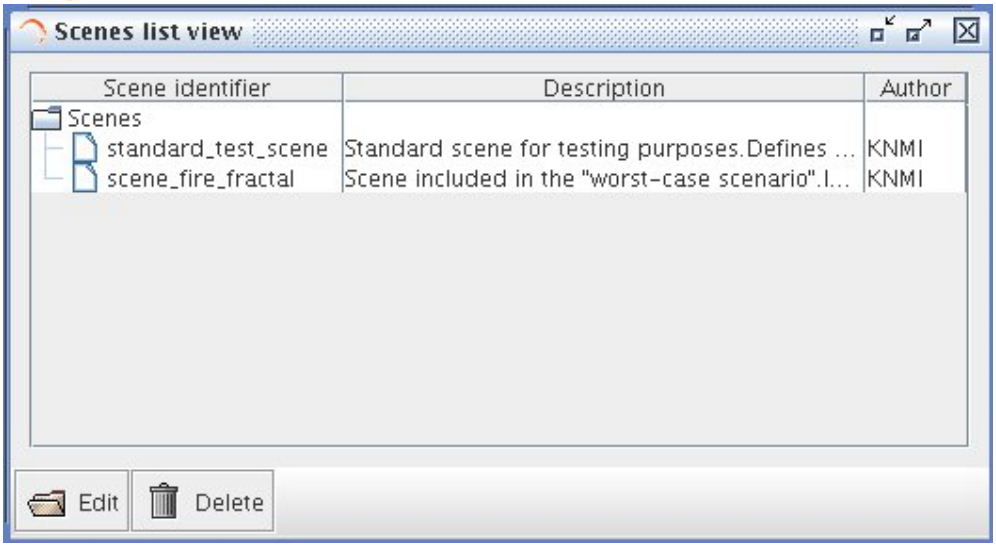


Figure 4-55: Scenes list view

The tree-like structure shows the name of each scene, a brief description and the name of the author.

Right-clicking over a scene will open the context-sensitive menu and double-clicking over it will open the scene. The tool bar in the bottom part of the frame grants access to the same functionalities of the pop-up menu.

4.11.2. New scene

Creating a new scene from the scratch is a simple process within ECSIM. The information that defines a scene is grouped in five different tabs: general data, scene dimensions and resolutions, atmosphere composition, surfaces and clouds and aerosols.

4.11.2.1. General data

In this tab users can define the following characteristics:

Identifier	Format	Purpose	Sample
Identifier	Long string	The name of the scene. This identifier must be unique in the scenes directory because it is also the name under which shall be stored as an XML file. The complete name of the file is shown in the right-most text field "File name". This file name must be validated by the operating system.	"scene_1"
Description	Long string	This is a brief description of the scene. Pressing control + enter will insert a line break in the text.	"This is a scene showing good sky conditions over Dublin"
Author	Medium string	The name of the author.	"DMS"

© DEIMOS Space S.L., 2007

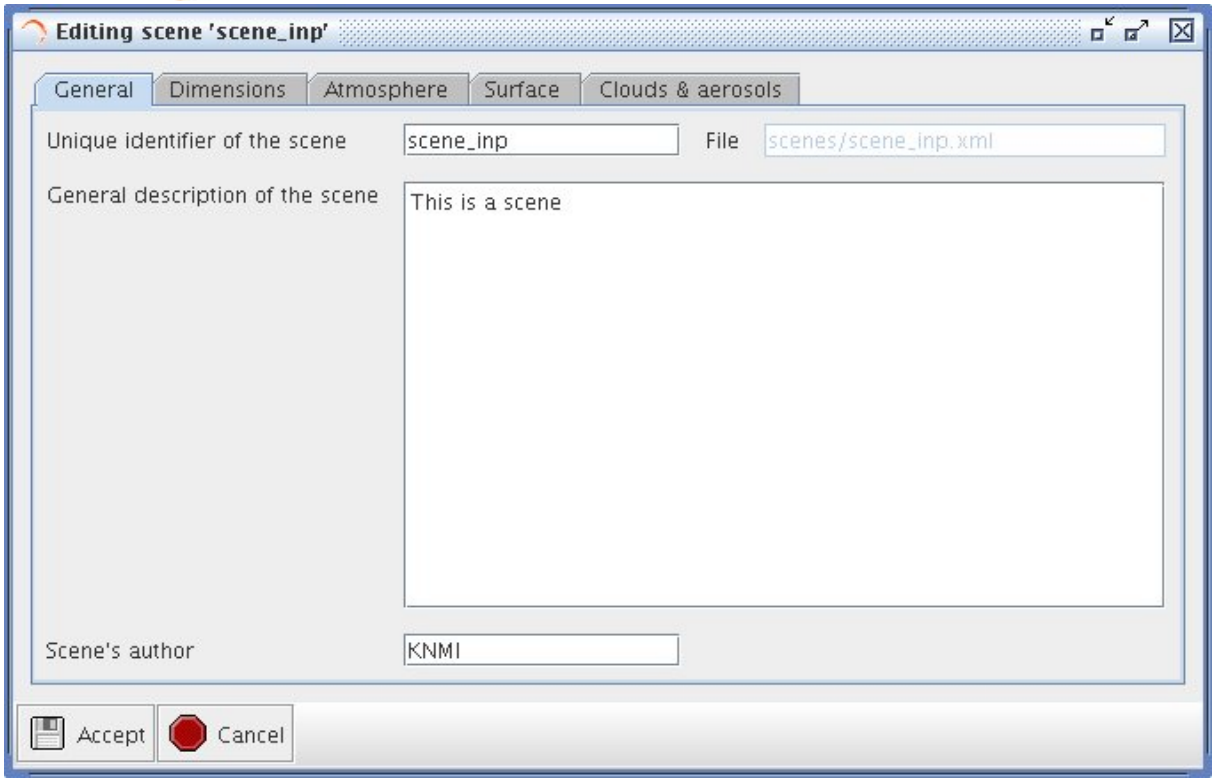


Figure 4-56: Scene. General data

4.11.2.2. Scene dimensions

In this frame users can define the X, Y and Z extensions of the scene and also specify the horizontal and vertical resolutions of the scene.

The horizontal resolution is fixed for the entire scene but it is possible to separate the vertical resolutions in different layers, managing the table shown in Figure 4-58.

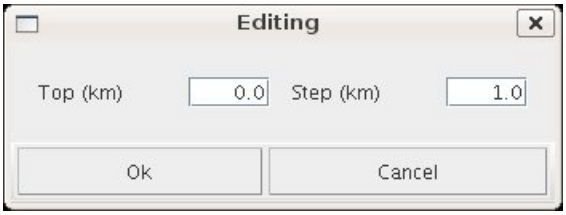


Figure 4-57: Scene. Vertical resolution

Clicking in the "add" or "remove" buttons marked with a document and a trash bin signs will, respectively, create a new vertical resolution layer or delete the selected one. Once the "add" button is pressed a dialog will come up and the user can introduce this data:

Attribute name	Format	Purpose	Sample
Top	Float	Upper border of the resolution layer	2.5
Step	Float	Local resolution of the layer. This must be less than the difference between its top and the top of the previous layer.	0.1

Double-clicking a resolution will also show this dialog and right-clicking over it will show a context-menu to edit the layer, add a new layer or remove the selected one.

© DEIMOS Space S.L., 2007

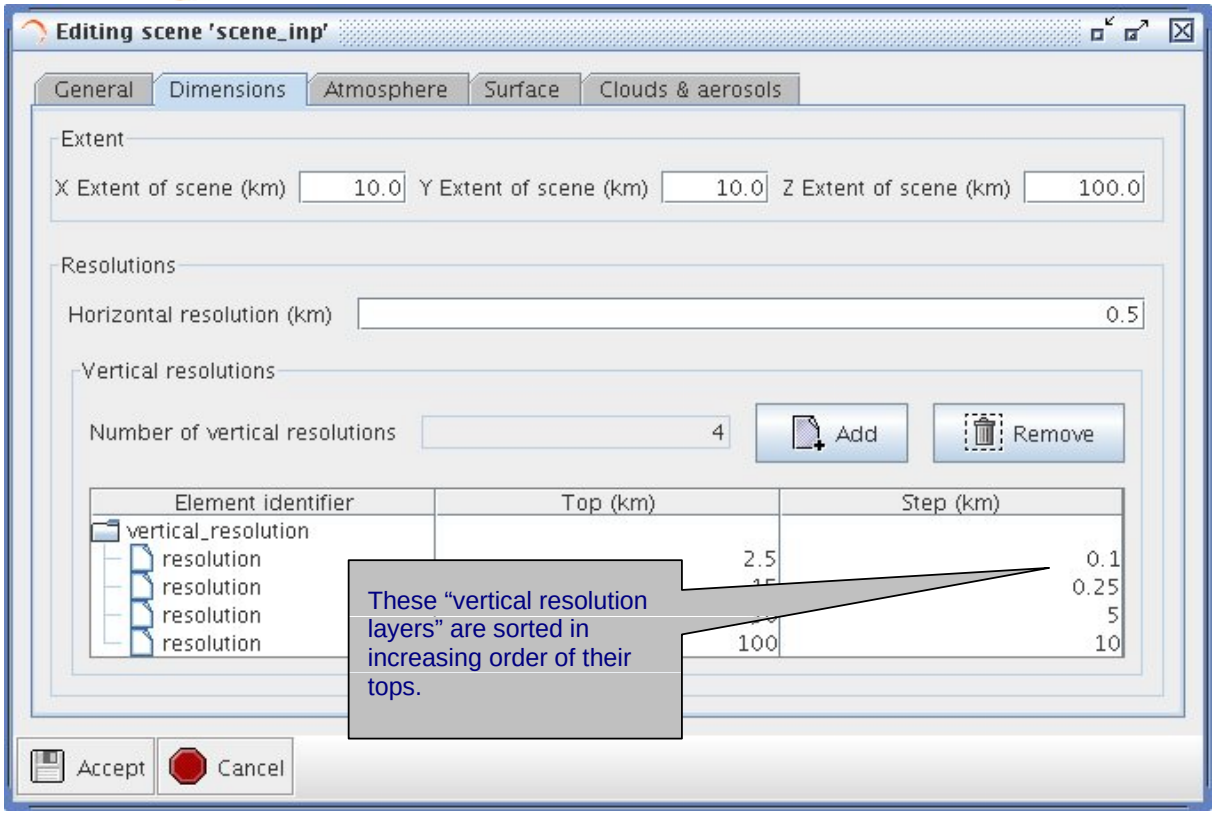


Figure 4-58: Scene. Dimensions and resolutions

4.11.2.3. Atmosphere composition

In this frame users can define the atmospheric gasses composition of the scene. Users can specify some common gasses and even include series of extra gasses.

As seen in Figure 4-59, users can define the following attributes:

Attribute name	Format	Purpose	Sample
Temperature-pressure profile.	File	An external file describing the composition of the atmosphere. This file must follow the [RD ICD] directives. Users can find a selection of profiles in $ECSIM_HOME/models/aux/standard_atmos_profiles	"Mid-latitude_summer.xml"
CO_2 mixing ratio.	Float	Includes the CO_2 in the calculations	350.0
H_2O scale factor.	Float	Includes the H_2O in the calculations	1.0
O_3 scale factor.	Float	Includes the O_3 in the calculations	1.0

© DEIMOS Space S.L., 2007

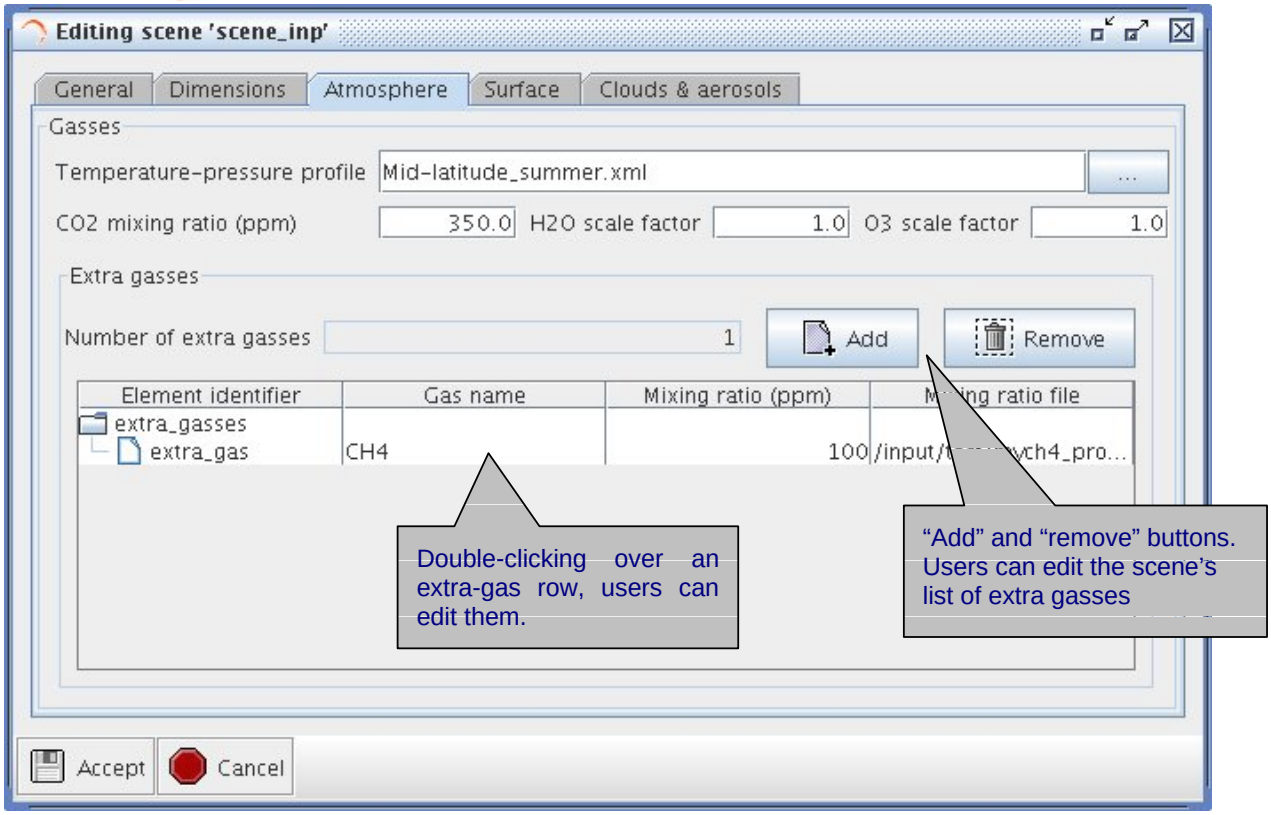


Figure 4-59: Scene. Atmosphere composition

- Extra gasses. It is possible to include more gasses (besides the three of the default set). Clicking the plus button will show a dialog to input information of a new gas. These are the attributes:

Attribute name	Format	Purpose	Sample
Gas name.	Short string	Descriptive name of the gas.	"CH4"
Mixing ratio.	Float	Mixing ratio of the gas.	1.0
Mixing ratio file	File	An external file containing the mixing ratio profile. This file must follow the [RD ICD] directives.	ch4_profile

Figure 4-60: Scene. Extra gas

© DEIMOS Space S.L., 2007

Clicking the trash bin button will erase the selected gas. Double-clicking over a gas row will also show the dialog. Right-clicking over it will show a context menu with the options of editing it, adding a new one or removing the selected gas.

4.11.2.4. Scene surfaces

In this frame users can determine the surface parameters for a three-dimensional scene. Users can specify a default ground type (present where no other surface region is defined) and a list of surface regions.

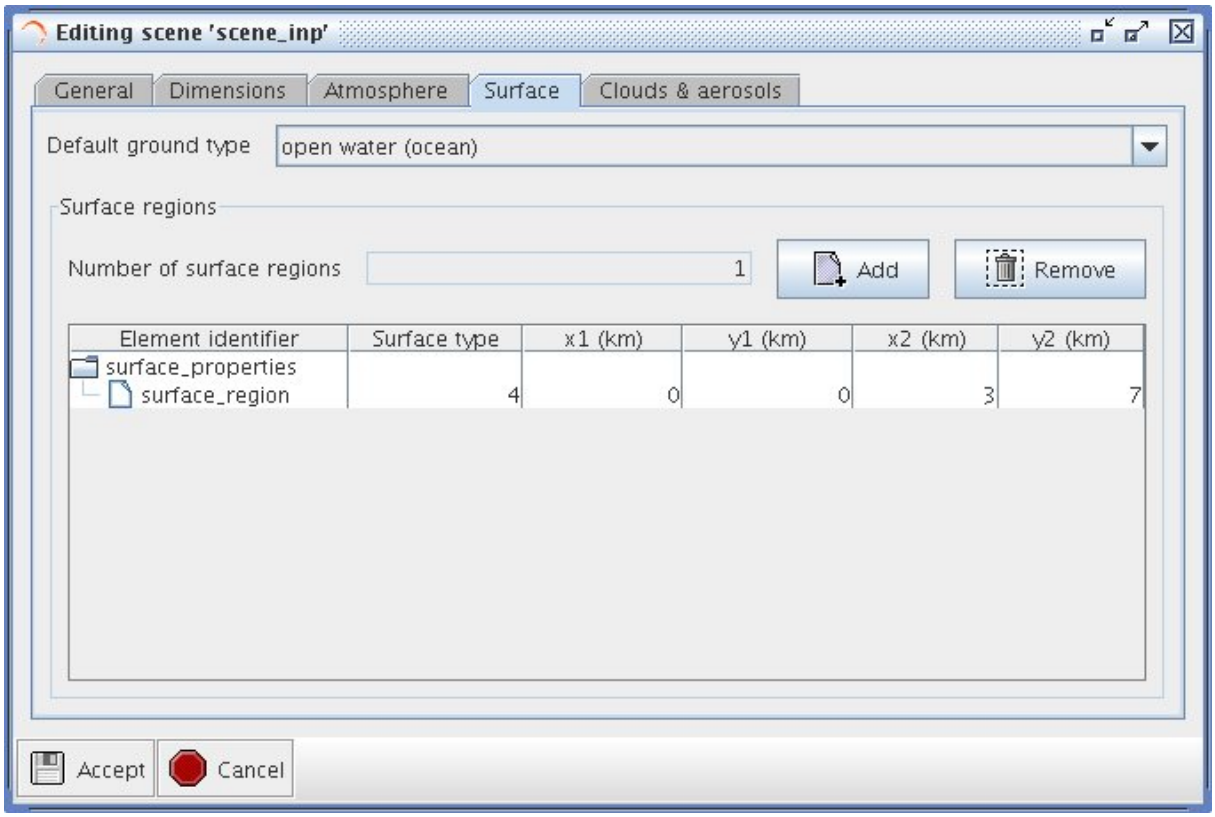


Figure 4-61: Scene – Surfaces

Figure 4-61 shows the ability to manage the "ground" part of the scene. Users can define the following attributes:

- Default ground type. This is the "general" type for the ground. Those parts of the scene's surface where a surface region is not defined will be of this type. The list of possible ground types is:
  - open water (ocean)
  - bright desert
  - dark desert
  - low-moderate vegation
  - moderate-high vegation
  - snow
- Surface regions. It is possible to define individual surface regions of a different type than the default. Clicking the plus button will show a dialog where the user can specify the information of a new surface region:

© DEIMOS Space S.L., 2007

Attribute name	Format	Purpose	Sample
Surface type	Options	The user can select a type from the list in the combo box component.	"bright desert"
Surface limits	Float	Users can specify the two-dimensional limits (x1, y1, x2, y2) of the region. They must be smaller than the general scene dimensions.	0.0 0.0 1.0 2.0

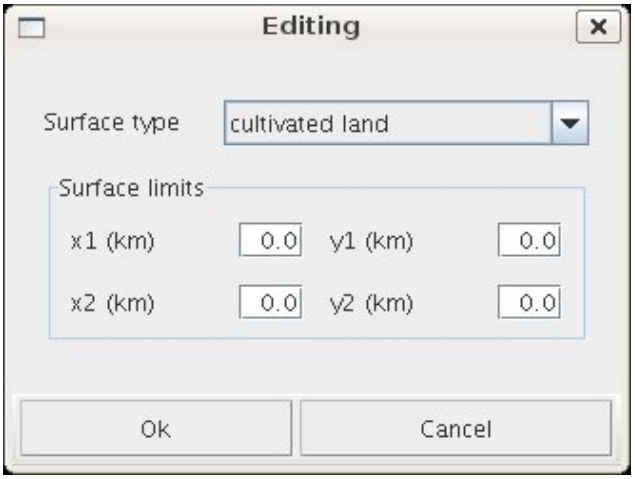


Figure 4-62: Scene. Surface region

Clicking the trash bin button will erase the selected surface region. Double-clicking over a surface row will also show the dialog. Right-clicking over it will show a context menu with the options of editing it, adding a new one or removing the selected surface region.

© DEIMOS Space S.L., 2007

4.11.2.5. Clouds and aerosols

Now it is time to define the main part of a scene: clouds and aerosols composition. Figure 4-63 shows that this is done by defining "scattering" regions with certain characteristics.

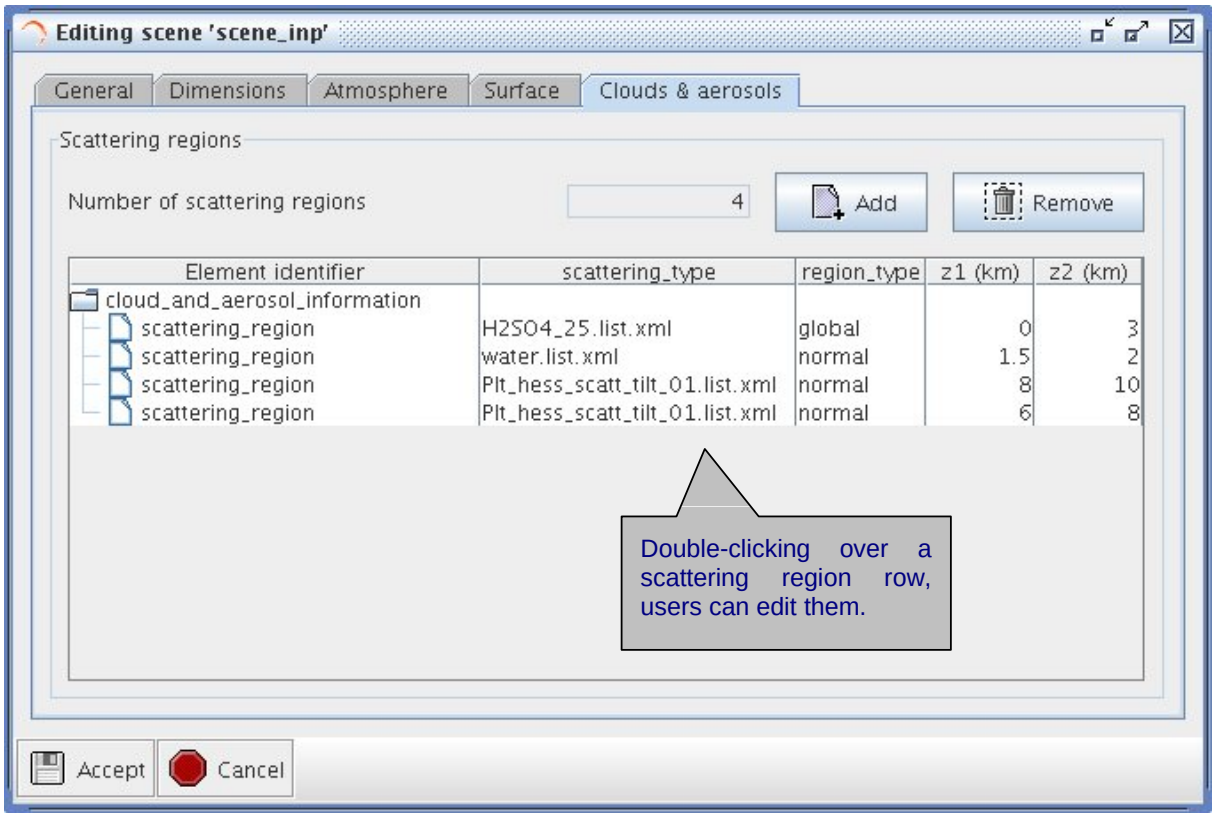


Figure 4-63: Scene. Clouds and aerosols

This table of scattering region allows the usual editing, adding and removing operations (double-clicks will edit it and a right-click will open a context menu).

Clicking the "Add" button will show a dialog where users can specify the information of a new scattering region as shown in Figure 4-64.

© DEIMOS Space S.L., 2007

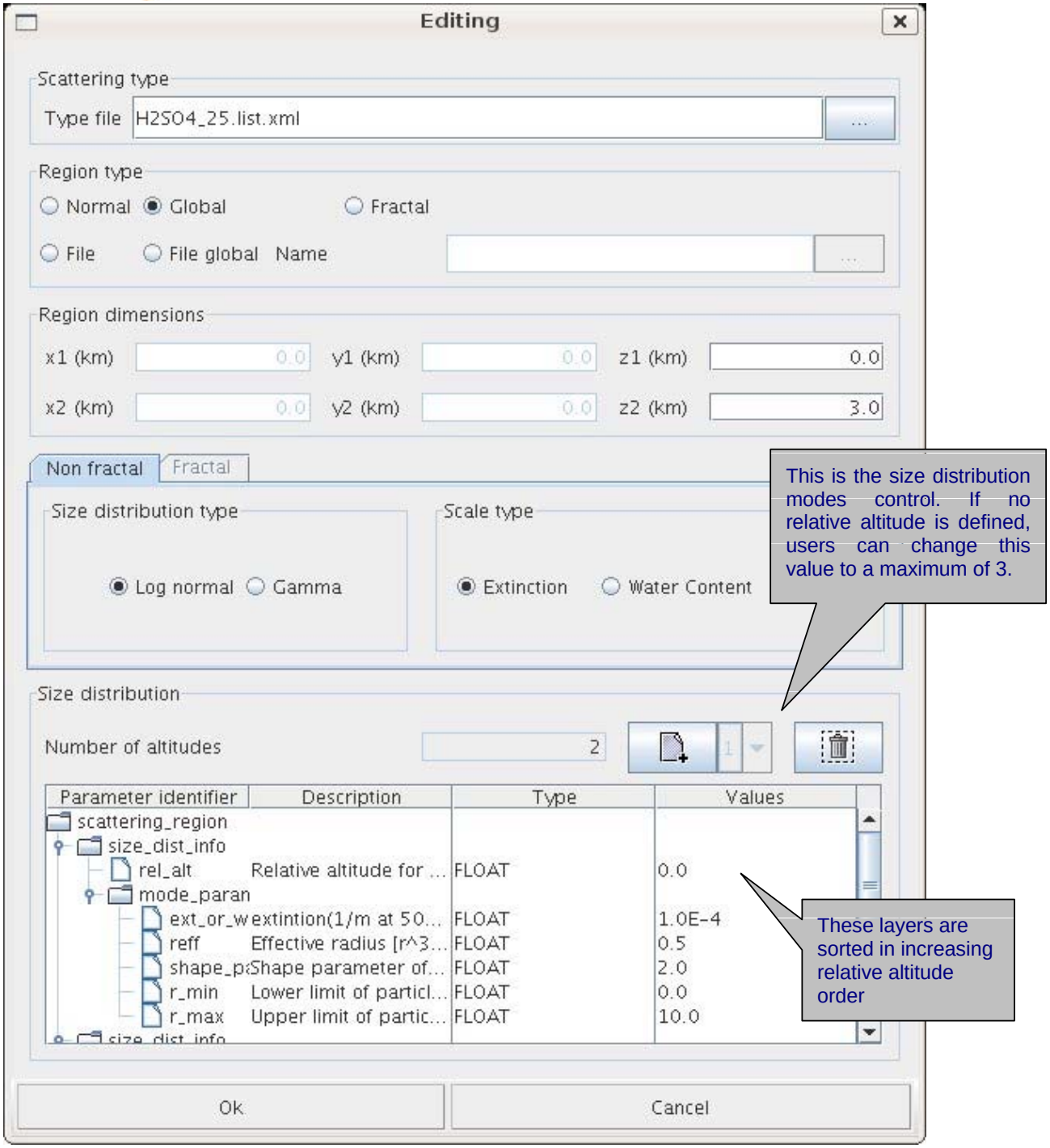


Figure 4-64: Scene. Non fractal scattering region

4.11.2.5.1. Common region attributes

- Scattering type. File type. Users can introduce the location of a file (or browse the directories to locate it) describing the type of scattering region. This external file must follow the directives marked in [RD ICD].

© DEIMOS Space S.L., 2007

- Region type. Users can choose one of these multiple options:
  - Normal. A normal region is a scattering region delimited with two terns (x, y and z) of coordinates.
  - Global. A global region is a scattering region that occupies the entire horizontal extension of the scene, thus only the Z limits need to be provided.
  - Fractal. Users can define fractal region if the CloudGen component is installed (see section 3.3.5).
  - File. This scattering region is going to be defined via an external file (that fulfils the [RD ICD]) and the usual two terns of coordinates. The location of this external file must be input in the text box or via the file chooser activated by the correspondent button.
  - File global. This scattering region is defined via an external file (that fulfils the [RD ICD]) and the limits of the scene in the Z axis. The location of this external file must be input in the text box besides or via the file chooser activated by the correspondent button.
- Region dimensions. Depending on the defined region type, users can input two or three-dimensional limits for the scattering region.

4.11.2.5.2. Non fractal region attributes

Attribute name	Format	Purpose	Sample
Size distribution type	Options: log normal or gamma	This is the type of the particle size distribution	gamma
Scale type	Options: extinction and water content.	This is the way the particle scale is measured.	extinction

- Relative altitudes. It is possible to define information about the size distribution of this scattering region. Users can add or remove individual size distributions with the "plus" and "minus" buttons. Users now can access to these individual parameters:

Attribute name	Format	Purpose	Sample
Relative altitude	Float (0.0 – 1.0)	Relative altitude in km for this entry	0.0
Mode - ext_or_wc	Float	Scale value. 1/m at 500nm if scale type is extinction or g/m^3 if water content	0.0001
Mode - reff	Float	Effective radius in microns. r^3/r^2 where r=D/2 and D is the particle maximum dimension	0.5
Mode – shape_parameter	Float	Shape parameter of size distribution	2.0
Mode – r_min	Float	Lower limit of particle sizes to include in microns	0.0
Mode – r_max	Float	Upper limit of particle sizes to include in microns	10.0

Users can define (before adding a relative altitude) how are going to be defined the size distributions of this scattering region, by changing the number of its modes.

© DEIMOS Space S.L., 2007

4.11.2.5.3. Fractal region attributes

Attribute name	Format	Purpose	Sample
Cloud generator	File	Path and call to the external cloudgen program	"usr/local/cloudgen-1.3/cloudgen"
Output file	File	Output file name of the cloudgen (netcdf) run	"iwc.nc"
Missing value	Float	The missing value used in the fractal code.	-999.0
Seed	Signed integer	Seed used by the random generator in the fractal code (negative numbers use time)	2
Threshold	Float	The IWC threshold defining the cloud boundaries in g/m^3	0.001
Generating level	Float	Level at which the cloud is generated, usually near cloud top in km.	9.5
Vertical exponent	Float	Exponent of the power spectrum in the vertical direction	-2.0
Outer scale	Float	The power spectrum becomes flat at scales larger than outer scale. In km.	50.0
Lower limit	Float	Lower limit of particle sizes to include. In microns.	0.0
Upper limit	Float	Upper limit of particle sizes to include. In microns	2000.0
Size correlation	Float	The correlation coefficient between effective radius and the logarithm of ice water content	0.5
Mixing	Unsigned integer (0 – 1)	The mixing represented by a change in the horizontal exponent may be anisotropy (0: no anisotropy, 1: smoother in fallstreak orientation)	0

- Relative altitudes. It is possible to define information about the size distribution of this scattering region. Users can add or remove individual size distributions with the "plus" and "minus" buttons. Users now can access to these individual parameters:

Attribute name	Format	Purpose	Sample
rel_alt	Float (0.0 – 1.0)	Relative altitude in km for this entry	0.0
fall_speed	Float	Particle fall speed at the relative altitude. In m/s	1.0

© DEIMOS Space S.L., 2007

Attribute name	Format	Purpose	Sample
horizontal_exponent	Float	The height variation of the exponent of the horizontal power.	-1.667
v_wind	Float	V wind velocity at the relative height. Used to overwrite model winds when velocity_overwrite='1'. In m s^{-1}	10.0
u_wind	Float	U wind velocity at the relative height. Used to overwrite model winds when velocity_overwrite='1'. In m s^{-1}	5.0
mode – wc_mean	Float	Mean water content. In g/m^3	0.0001
Mode – std_dev	Float	Standard deviation of the natural logarithm of the water content. In g/m^3	2.0
Mode – reff_mean	Float	Mean effective radius log_normal distribution. In microns.	70.0
Mode – reff_std_dev	Float	Standard deviation of the natural logarithm of the effective radius log_normal distribution. In microns.	0.2

© DEIMOS Space S.L., 2007

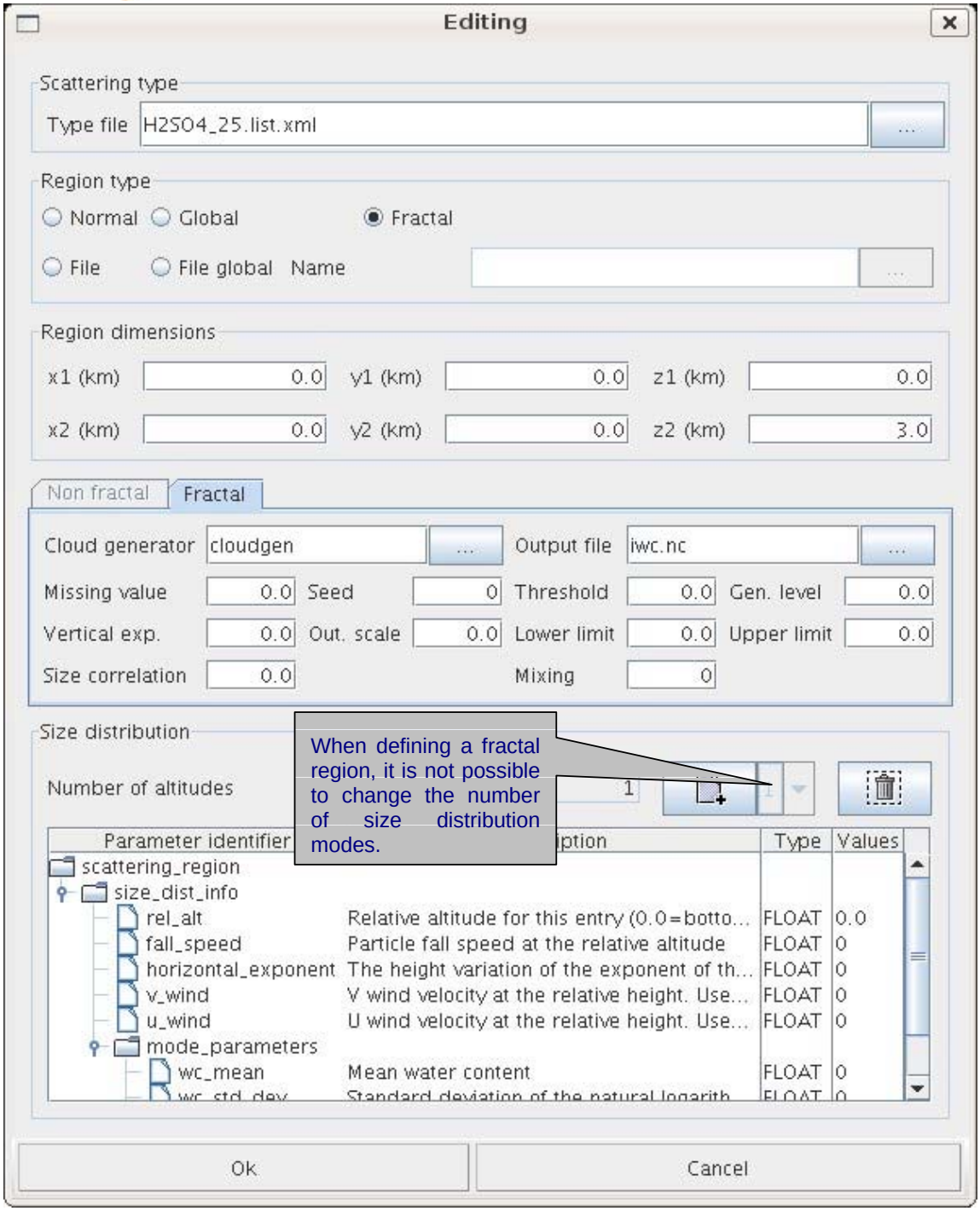


Figure 4-65: Scene. Fractal scattering region

4.11.3. Delete scene

Once users have selected a certain scene and the "delete scene" action, a confirmation dialog asking will come up. Once the user is sure to proceed with the deletion, the XML file that describes the scene is erased from the file system.

© DEIMOS Space S.L., 2007

4.11.4. Compare scenes

This functionality lets the user graphically compare two given scenes.

As shown in Figure 4-66 users can select two scenes to display their graphical representations in 3D.

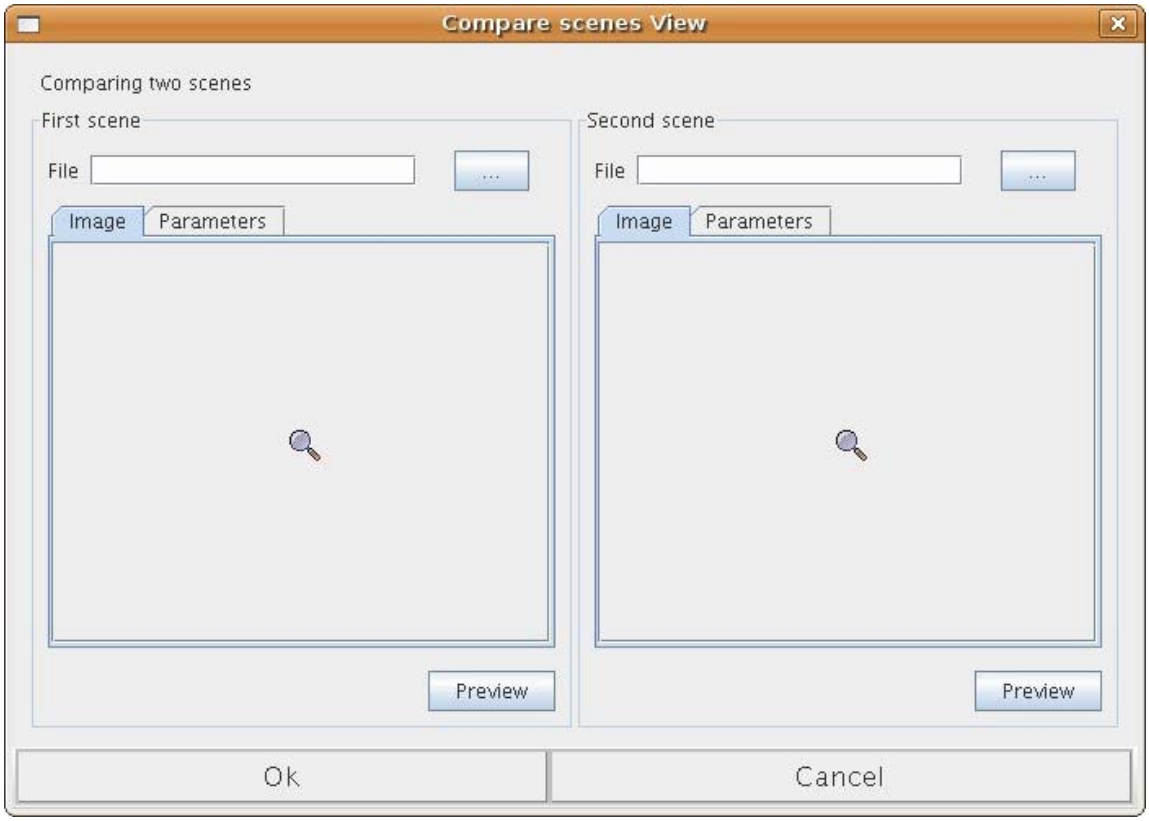


Figure 4-66: Scene Comparison

4.11.5. Import scene

ECSIM can import scene information from third-party applications.

Basically, when creating a new scene, you can define a scattering region as a "read from file" region then a "size distribution file" must be specified. Here users can select a file formatted as the size distribution files generated by lw_msi_lid_rad_l2b_3d as described in [RD ICD].

4.11.6. Merge scene

This functionality consists of creating a third scene from two independent scenes. The generated scene will sum the scene's dimensions and will contain every scattering region defined in both input scenes.

Accessing to this functionality will present the frame shown in Figure 4-67.

© DEIMOS Space S.L., 2007

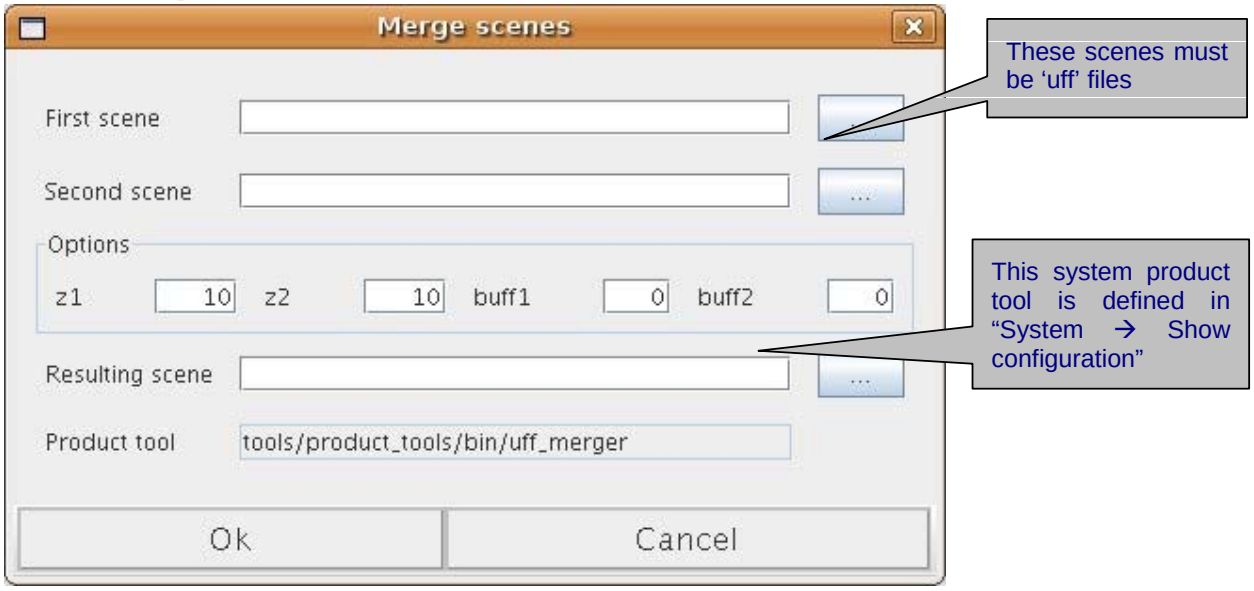


Figure 4-67: Scene. Merge

Users can input the location and name of the input scenes (first and second) and provide the name and location of a resulting scene (combination of the two others). Once finished, clicking on the "Ok" button will call the designated product tool to perform the merge operation.

4.12. Executions

As mentioned in section 4.6 ECSIM can be divided in three logical parts: repository, scenes and executions.

The third of these parts, named executions, represents the dynamic view of the system. Here the executed sessions are stored with their input and output data.

Users can consult their results and log messages generated, as well as re-run sessions as needed.

4.13. Results

Once a session is executed, an execution result (whether it was successful or not) is stored into the system into the <sessions> folder, named as "<session_id>.<starting_time>". Starting time is coded as "YYYYMMDDHHmmSSsss"[9] in local time.

[9] "sss" denotes miliseconds

© DEIMOS Space S.L., 2007

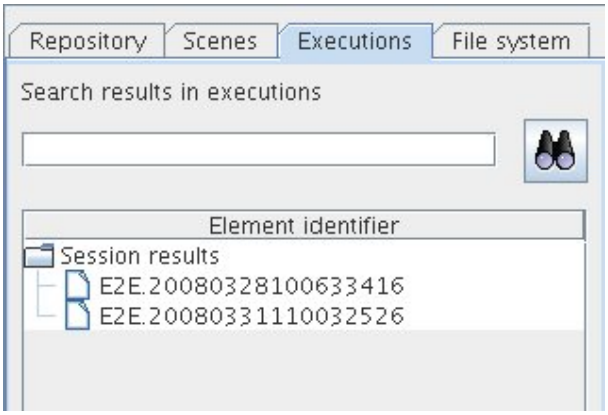


Figure 4-68: Side bar. Executions

Users can access to the list of results known by the system in the executions view of the side bar or via the "Executions" menu (Figure 4-69) from the main menu.

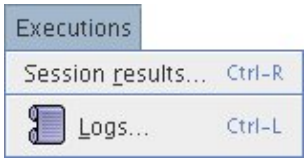


Figure 4-69: Results menu

Figure 4-70: Results pop-up menu

Operations involving results include the following:

- *List* – present the list of existing execution results;
- *View* – consult the data of an existing execution result;
- *Re-run* – Starts a new session execution. This new session is a replica of the former, but shall create a new session folder.
- *Report generation* – shows a text report describing the execution.
- *Deletion* – delete an execution result from the system.

4.13.1. Result view

Accessing this functionality, users can consult the result of a session execution. Figure 4-71 shows a window very similar to a session editor but including more information.

© DEIMOS Space S.L., 2007

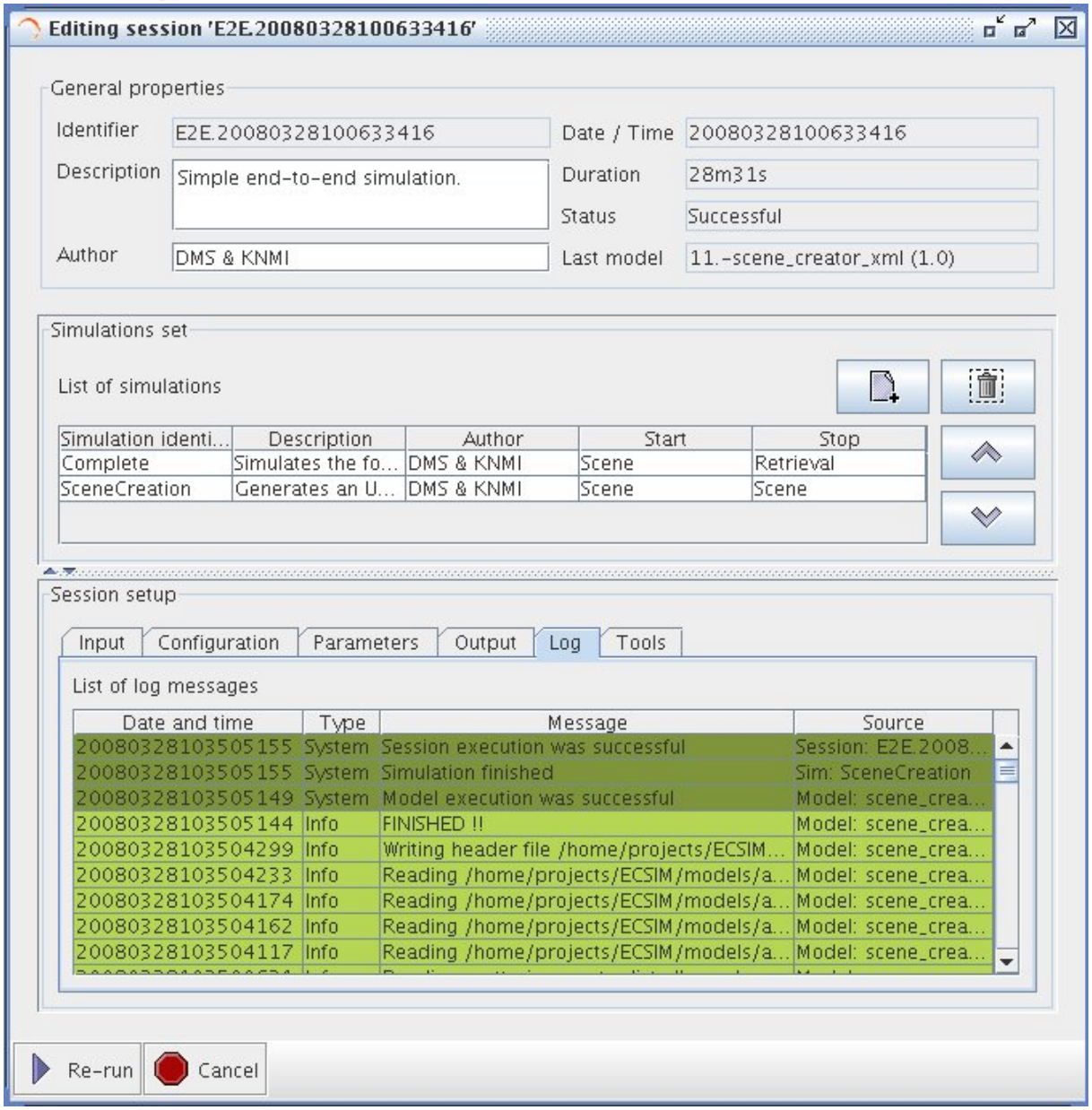


Figure 4-71: Execution results

First of all, some data is presented in the "general properties" region showing this attributes:

- Date / time – this is the local computer date and time that the execution began. This date and time can also be part of the simulation identifier to distinguish this session execution from others;
- Duration – the time (in minutes and seconds) elapsed from the starting time until the execution was finished or interrupted;
- Status – the overall status of the execution. The possible values are "Failed, "Successful" and "Aborted";

© DEIMOS Space S.L., 2007

- Last model – This is the number and identifier of the last model successfully executed. In case of a successful execution, this model must coincide with the very last model of the session. This information is useful for the user to know which model was erroneous.

Second, the Log tab is filled with the log messages generated by the session execution. Users can access all these messages to check its performance.

4.13.2. Result re-run

Accessing to this functionality, users can repeat the execution of a previously executed session. If the session execution was successful, the system just creates another execution (changing the starting date and time) but, if the previous execution was aborted or failed, the system will inform the user with the dialog shown below.

Session execution
This session has been already executed and it had a failure.
Do you want to restart or resume it starting from last valid model?
Restart
Resume

Figure 4-72: Result. Re-run

Users now can choose to restart the execution from the beginning or try to resume the execution, that is, to continue the execution from the last valid model. So the execution will continue provided that the outer conditions that ruined the previous run have been corrected.

4.13.3. Report generation

This functionality is accessed by clicking ion the "Generate report …" option from the Results pop-up menu. A window similar to the one shown in Figure 4-73 is presented to the user.

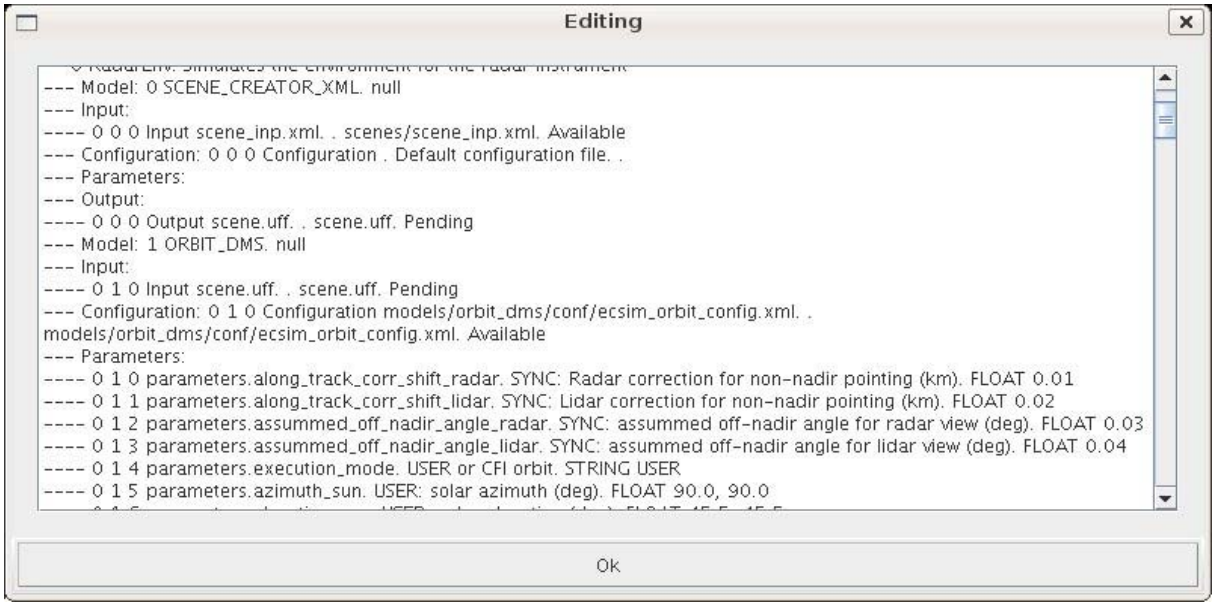

Editing
--- Model: 0 SCENE_CREATOR_XML. null
--- Input:
---- 0 0 0 Input scene_inp.xml. . scenes/scene_inp.xml. Available
--- Configuration: 0 0 0 Configuration . Default configuration file. .
--- Parameters:
--- Output:
---- 0 0 0 Output scene.uff. . scene.uff. Pending
--- Model: 1 ORBIT_DMS. null
--- Input:
---- 0 1 0 Input scene.uff. . scene.uff. Pending
--- Configuration: 0 1 0 Configuration models/orbit_dms/conf/ecsim_orbit_config.xml. .
models/orbit_dms/conf/ecsim_orbit_config.xml. Available
--- Parameters:
---- 0 1 0 parameters.along_track_corr_shift_radar. SYNC: Radar correction for non-nadir pointing (km). FLOAT 0.01
---- 0 1 1 parameters.along_track_corr_shift_lidar. SYNC: Lidar correction for non-nadir pointing (km). FLOAT 0.02
---- 0 1 2 parameters.assummed_off_nadir_angle_radar. SYNC: assummed off-nadir angle for radar view (deg). FLOAT 0.03
---- 0 1 3 parameters.assummed_off_nadir_angle_lidar. SYNC: assummed off-nadir angle for lidar view (deg). FLOAT 0.04
---- 0 1 4 parameters.execution_mode. USER or CFI orbit. STRING USER
---- 0 1 5 parameters.azimuth_sun. USER: solar azimuth (deg). FLOAT 90.0, 90.0
Ok

Figure 4-73: Execution report

© DEIMOS Space S.L., 2007

This execution report consists in a textual description of the same data that users can access with the "Result view" functionality. The only difference resides in that this textual information can be copied and pasted into another application outside the ECSIM system.

4.13.4. Delete result

Users can select a certain execution result and choose the option to delete it. Once users confirm the operation the execution result is erased from the repository and the file system. Log messages associated with this session execution result will also be erased.

4.14. Logs

Executing sessions has a secondary consequence: a set of events are produced and stored for the system. These events are described in detail in section 4.10.5.

Users can access to the complete set of logs stored by the system in the "Logs" menu (Figure 4-74) from the main menu.

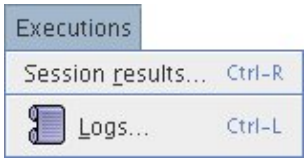


Figure 4-74: Logs menu

4.14.1. Log messages list

The figure below shows a window with a list of log messages stored by the system. As can be seen, the table shows the computer date and time when the platform intercepted the event, the type of the event, a message describing the event, the identifier of the session associated to the event and its detailed source (model, simulation, session or system).

This list of events is sorted (by default) in increasing time order until filling the "Maximum number of rows displayed" field. Users can change the number of log messages to be displayed. For example, if the "Maximum number of rows displayed" is set to 10, the list shall display the last 10 messages of the log session.

Users can also filter this list. Users can select a field, input a string that must contain this field and press the "filter" button. Then another search is performed into the system and the results are shown on screen. Only records fulfilling the filter restriction will be shown.

© DEIMOS Space S.L., 2007

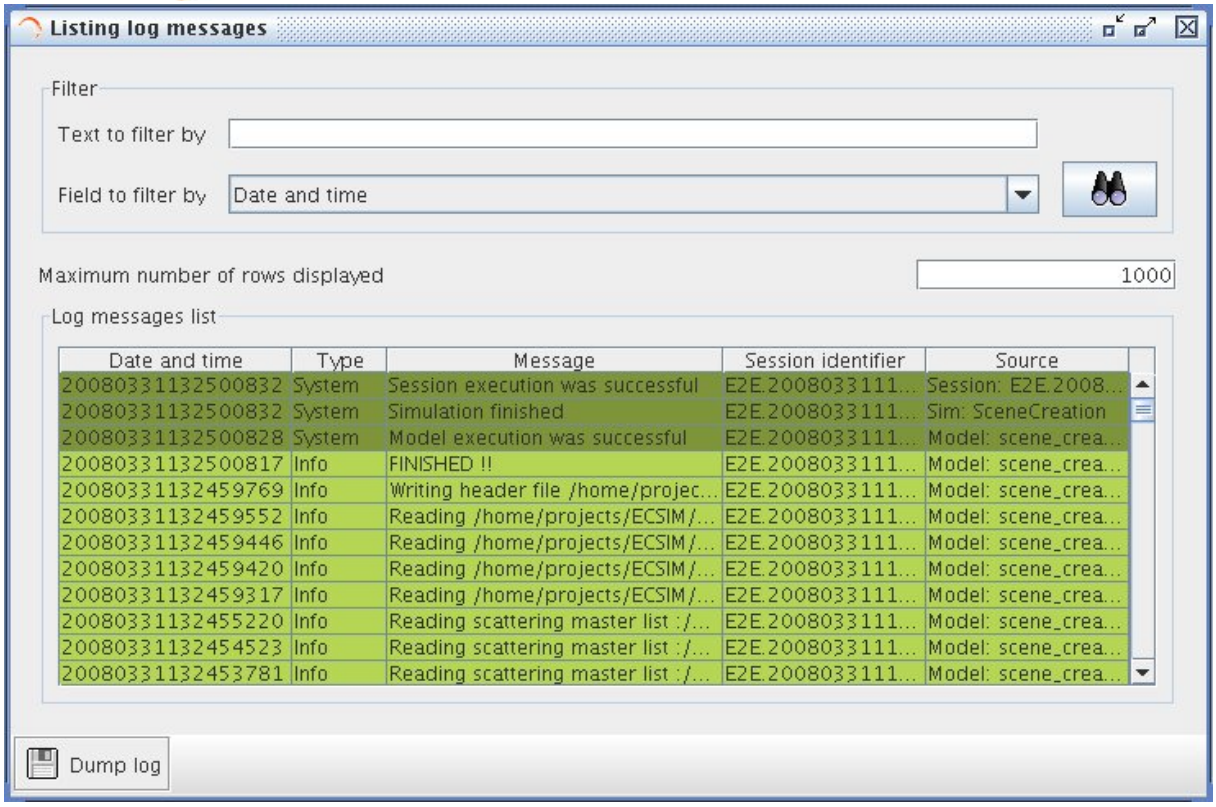

Listing log messages
Filter
Text to filter by
Field to filter by
Date and time
Maximum number of rows displayed
1000
Log messages list
Date and time
Type
Message
Session identifier
Source
20080331132500832 System Session execution was successful E2E.2008033111... Session: E2E.2008...
20080331132500832 System Simulation finished E2E.2008033111... Sim: SceneCreation
20080331132500828 System Model execution was successful E2E.2008033111... Model: scene_crea...
20080331132500817 Info FINISHED !! E2E.2008033111... Model: scene_crea...
20080331132459769 Info Writing header file /home/projec... E2E.2008033111... Model: scene_crea...
20080331132459552 Info Reading /home/projects/ECSIM/... E2E.2008033111... Model: scene_crea...
20080331132459446 Info Reading /home/projects/ECSIM/... E2E.2008033111... Model: scene_crea...
20080331132459420 Info Reading /home/projects/ECSIM/... E2E.2008033111... Model: scene_crea...
20080331132459317 Info Reading /home/projects/ECSIM/... E2E.2008033111... Model: scene_crea...
20080331132455220 Info Reading scattering master list :/... E2E.2008033111... Model: scene_crea...
20080331132454523 Info Reading scattering master list :/... E2E.2008033111... Model: scene_crea...
20080331132453781 Info Reading scattering master list :/... E2E.2008033111... Model: scene_crea...
Dump log

Figure 4-75: Logs list view

Moreover, users can access the "dump log" functionality at the bottom of the window. Once selected, users can select the name and location of the log destination file. Then, the list of logs shown by the window is stored in the file system.

4.15. File system

The last tab in the side-bar, named "File system" is a way to access the folder structure under the $ECSIM_HOME.

Organized in the tree-like structure, the user can easily locate every needed file.

This structure is refreshed every time an operation involving files is performed or once the user presses the "refresh" button.

© DEIMOS Space S.L., 2007

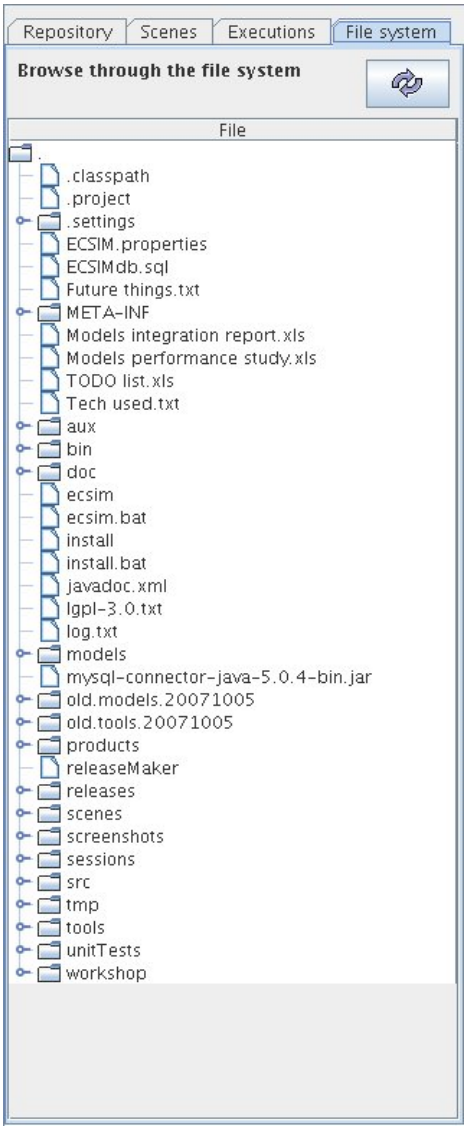


Figure 4-76: File system view

4.15.1. Tool execution

As declared in section 4.5.2, the ECSIM system can associate external tools to a series of file extensions.

In case the user right-clicks over a file name whose extension is associated to one or several product tools, a menu showing some actions will pop-up.

© DEIMOS Space S.L., 2007

Figure 4-77: IO file pop-up menu

Once you select the desired action, a dialog will show up asking you for completing the executable command line.

Users can accept the default parameters (the absolute path of the file) or can add extra. Users can also make use of the ECSIM environment variables (described in section 3.5.1) writing the dollar symbol and its name. For example, $ECSIM_HOME or $STD_ATMOS.

Once accepted the parameters, the external program will be executed in a separate thread (so the ECSIM operations are not interrupted).

4.16. Persistent storage - Database and file system

Most information systems must store information in a persistent way. ECSIM system trusts in a relational database to store structural information and the file system to store the input/output/configuration files. The following list shows which ECSIM elements are stored in database and which into the file system.

Element	Storage
System. Configuration	File system. $ECSIM_HOME/ECSIM.properties
System. Tools	Database
Repository. Descriptors	Database
Repository. Models	Database
Repository. Simulations	Database
Repository. Sessions	Database
Repository. Session script	File system. <sessions_folder>/<name> /<name>.sh
Scenes. Scene description	File system. <scenes_folder>/<name>.xml

© DEIMOS Space S.L., 2007

Element	Storage
Scenes. Scene binary file	File system. ❑ <scenes_folder>/<name>.uff ❑ <scenes_folder>/header_<name>.uff ❑ <scenes_folder>/size_dist_#_<name>.uff
Executions. Results	Database
Executions. Logs	Database
Execution. Dumped log session	File system
Execution. Input/output/configuration files	File system <sessions_folder>/<name>/<index. simulation_name>/<filename>

4.16.1. Database maintenance

Currently, the database is allocated in a local MySQL server, named "ecsim" and can be accessed with a user named "ecsim" with password "ecsim". This user and password cannot be changed or the ECSIM system will not be able to access it and, consequently, it will not start.

The user (or the database server administrator) is responsible to regularly back-up, de-fragment, clean and perform similar maintenance operations to guarantee the database integrity. Users can execute the following script to perform a manual backup to the ECSIM database.

```
~/ECSIM$ mysqldump --user=ecsim --password=ecsim ecsim > ECSIMdb.bk.sql
```

In case of a major corruption problem or if the user would like to roll back to the original ECSIM database configuration, it is possible to call the installation script:

```
~/ECSIM$ sh install <user> <password>
```

Where <user> and <password> are the name and password of the MySQL database root user.

Alternatively, it is possible for the user to directly use the "ECSIMdb.sql" script included in $ECSIM_HOME this way:

```
~/ECSIM$ mysql --database=ecsim --user=ecsim --password=ecsim < ECSIMdb.sql
```

4.17. Table of acceleration keys

This section describes the list of "acceleration keys" accessible for the user. These combinations of keys, when pressed, access the desired platform functionality without using the mouse and menu systems.

Here is the complete list:

Module	Action	Acceleration key
System	Exit the system	Control + x
System	Exit the system	Alt + F4

© DEIMOS Space S.L., 2007

Module	Action	Acceleration key
System. Tools	Tool list	Control + t
System. Configuration	Show configuration	Control + o
System	About ECSIM	Control + a
Repository. Descriptors	Descriptor list	Control + d
Repository. Models	Model list	Control + m
Repository. Simulations	Simulation list	Control + i
Repository. Sessions	Session list	Control + s
Scenes	Scene list	Control + c
Executions. Results	Result list	Control + r
Executions. Logs	Log messages list	Control + l
Window	Close internal frame	Alt + z

© DEIMOS Space S.L., 2007

4.18. Error messages

ECSIM platform controls its correct behaviour with an error handling system. Users are informed about the nature of the error and a possible way to correct it.

Here is a list of different kinds of raised errors:

Module	Operation	Error	Comments
System. Configuration	Adding a new variable	Validation error	Follow the instructions to correct the value
System. Tools	Accepting changes	Tool addition failed	You have chosen a duplicated identifier. Please provide a different identifier
		Validation error	Follow the instructions to correct the value
	Deleting a tool	Database error	Possible database failure. Is your database running?
	Executing a tool	File IO error	Follow the instructions
Repository. Descriptors	Accepting changes	Descriptor modification failed	Possible database failure. Is your database running?
		Descriptor addition failed	You have chosen a duplicated identifier. Please provide another identifier.
		Validation error	Follow the instructions to correct the value
	Adding an IO file	Validation error	Follow the instructions to correct the value
		A descriptor shall not have associated two files with the same id	Please choose another identifier

© DEIMOS Space S.L., 2007

Module	Operation	Error	Comments
	Deleting a descriptor	Database error	Possible database failure. Is your database running?
Repository. Models	Accepting changes	Validation error	Follow the instructions to correct the value
		Model addition failed	You have chosen a duplicated identifier. Please provide a different identifier
	Deleting a model	Database error	Possible database failure. Is your database running and configured?
	Creating a new version	Database error	Possible database failure. Is your database running and configured?
Repository. Results	Deleting a result	Database error	Possible database failure. Is your database running and configured?
Repository. Scenes	Accepting changes	Scene addition failed	You have chosen a duplicated identifier. Please provide a different identifier
		Validation error	Follow the instructions to correct the value
	Adding a relative altitude to a scattering region	Validation error	Follow the instructions to correct the value
	Comparing scenes	File IO error	Follow the instructions
	Deleting a scene	File IO error	Follow the instructions
	Editing a vertical resolution layer	Vertical resolution (step) of this layer must be less than its size	Please input a valid step
	Editing a scene	File IO error	Follow the instructions
	Merging scenes	Incompatible scenes	You must select two valid scenes to merge
	New scene	File IO error	Follow the instructions

© DEIMOS Space S.L., 2007

Module	Operation	Error	Comments
	Previewing a scene	File IO error	Follow the instructions
Repository. Sessions	Accepting changes	Validation error	Follow the instructions to correct the value
		Database error	Possible database failure. Is your database running and configured?
		Session addition failed	You have chosen a duplicated identifier. Please provide a different identifier
	Adding a simulation	Session identifier cannot be void	Please provide a valid identifier before adding a simulation
	Deleting a session	Database error	Possible database failure. Is your database running and configured?
	Generating a script	File IO error	Follow the instructions
	Iterating parameters	Invalid list of values	Please input a comma-separated list of valid values (no blanks)
		Validation error	Follow the instructions to correct the value
	Removing a simulation	There is no simulation selected	Please select a simulation to remove
	Running a session	Cannot run an unnamed session	Please input a valid identification to the session
		Validation error	Follow the instructions to correct the value
		File IO Error	Follow the instructions
Repository. Simulations	Accepting changes	Simulation modification failed	Possible database failure. Is your database running and configured?

© DEIMOS Space S.L., 2007

Module	Operation	Error	Comments
		Simulation addition failed	You have chosen a duplicated identifier. Please provide a different identifier
		Validation error	Follow the instructions to correct the value
	Deleting a session	Database error	Possible database failure. Is your database running and configured?
	Executing a simulation	Validation error	Follow the instructions to correct the value
	Setting limits	Validation error	Follow the instructions to correct the value
Executions. Log	Dumping the log	File IO error	Follow the instructions

In general, every time an input value is needed, the platform will perform a validation process. If the input does not comply with the needed format, the user will be informed with a self-explained message.

Errors not shown as part of the graphical interface are not controlled messages, They correspond to messages from the standard output or error stream.

When executing a simulation, models raise their own error messages and they are intercepted by the system and shown as log messages in the execution view. Here is a list of possible kinds of errors raised by the models.

Model (or model type)	Error condition	General Error reported	Resolution
All forward, retrieval, and instrument models	Incorrectly formatted or missing command line arguments	Error and Usage (help) message is reported	Adjust command line arguments to match the usage statement.
	Ill-formatted xml configuration file	xml syntax error is reported along with line and column where parsing failed	Edit configuration file based on the reported error.

© DEIMOS Space S.L., 2007

Model (or model type)	Error condition	General Error reported	Resolution
	Mismatch between declared parameter type and dimensions and supplied parameter within the configuration file	Mismatch error is reported along with offending parameter statement.	Edit configuration file based on the reported error.
	Missing non-optional parameters	Name of missing parameter is reported	Edit configuration file based on the reported error (supply the missing parameter)
	Missing input data files or necessary auxiliary data files (including scattering libraries)	Name of missing data files are reported.	Verify that the relevant files have been installed and the ECSIM path environment variables have been correctly set.
	Unable to write output file	Writing failure is reported along with name and path of the relevant file	Verify that the specified output directory exist and that the user has write permission.
Plotting tools	Requested quantity is not present(with the expected number of dimensions) within the ncdf data file	Error message listing all available quantities for the supplied ncdf file is generated.	Rerun having made a choice amongst the available quantities.
	Unable to create the plot using the requested graphics device	Unformatted PGPLOT errors will be reported	Rerun using an available graphics device. To check the available devices specify /? as the device (must be done in command line mode). To add more devices PGPLOT must be recompiled and installed.

© DEIMOS Space S.L., 2007

Model (or model type)	Error condition	General Error reported	Resolution
L1_l2_compare	Resolution mismatch between the 2 data streams the user wishes to compare	An error message is generated.	Follow the instructions in the error message (first use l1_l2_rebin to harmonize the resolution of the 2 data files)
Uff_merger	Resolution mismatch between the 2 UFF files the user wishes to compare	An error message is generated.	Follow the instructions in the error message (first use uff_averager to harmonize the resolution of the 2 uff files)
Extract_quantity Extract_hor Etc..	Missing input data files or necessary auxiliary data files (including scattering libraries)	Name of missing data files are reported.	Verify that the relevant files have been installed and the ECSIM path environment variables have been correctly set.
	Unable to write output file	2. Writing failure is reported along with name and path of the relevant file	Verify that the specified output directory exist and that the user has write permission.

© DEIMOS Space S.L., 2007

5. ECSIM – OPERATIONS MANUAL

This chapter details some examples of procedures on the ECSIM system. Experienced users can skip its reading.

5.1. Forward branch simulation

5.1.1. Operation objectives

The objective is to perform a simulation of the forward branch of all the instruments. The simulation starts creating a scene and ends simulating the CPR instrument.

As a result of this simulation we can visually compare how the reflectivity of the original scene was and how the instrument sensed it.

5.1.2. Pre-conditions

A previously defined session called "Forward" exists in the repository. This session includes:

- One simulation also called "Forward" that concatenates the scene creator, orbital, "rad_filter" and the "radar" models.
- A list of product tools to show to visualize the results. These tools must be well defined and accessible for the system.

The following set of input, auxiliary and configuration files will be defined:

Model and type	File name	File instance
scene_creator_xml. Input	scene_inp.xml	$ECSIM_HOME/sessions/Forward/0-Forward/scene_inp.xml
orbit_dms. Input	scene.uff	Pending
orbit_dms. Configuration	orbit_dms_config.xml	$ECSIM_HOME/sessions/Forward/0-Forward/orbit_dms_config.xml
rad_filter. Input	uff_adjust.xml	$ECSIM_HOME/products/uff_adjust.xml
rad_filter. Input	scene.uff	Pending
rad_filter. Input	orbit.xml	Pending
rad_filter. Configuration	rad_filter_config.xml	$ECSIM_HOME/sessions/Forward/0-Forward/rad_filter_config.xml
radar. Input	rad_filter_out.nc	Pending
radar. Input	orbit.xml	Pending
radar. Configuration	radar_config.xml	$ECSIM_HOME/sessions/Forward/0-Forward/radar_config.xml

© DEIMOS Space S.L., 2007

5.1.3. Steps

- Select from the repository tab of the side bar, a session named "Forward". Right-click on it and select the "Edit" option.
- Browse every field of the input, configuration, parameters, output and tools tabs to edit the values you desire. You can use the files in the table above to perform a sample full end-to-end simulation.
- Once everything is as your wishes, click on the "Run" button to start the execution.
- The execution will start and you can see its progress status.
- Once the simulation ends, all the post-processing tools will start and you can see a graphical representation of how the instrument senses the scene and the generated orbital file.
- Other operations you can do upon the execution's end are to check every other generated file or the final execution report where you can find detailed information about it.

5.2. Single Instrument retrieval

5.2.1. Operation objectives

The objective is to perform a simulation of a single instrument and to retrieve its output. In this case, it will simulate the behaviour of the ATLID. Finally, a graphical comparison between the original and the reconstructed scene is shown.

5.2.2. Pre-conditions

A previously defined session called "Lidar" exists in the repository. This session includes:

- One simulation, named "Lidar". This simulation starts from the scene creator to the retrieval stage and it contains only models related with the ATLID instrument.
- A list of product tools to visualize the reconstructed scene.

The following set of input, auxiliary and configuration files will be defined:

Model and type	File name	File instance
scene_creator_xml. Input	scene_inp.xml	$ECSIM_HOME/sessions/Lidar/0-Lidar/scene_inp.xml
orbit_dms. Input	scene.uff	Pending
orbit_dms. Configuration	orbit_dms_config.xml	$ECSIM_HOME/sessions/Lidar/0-Lidar/orbit_dms_config.xml
lid_filter. Input	uff_adjust.xml	$ECSIM_HOME/products/uff_adjust.xml
lid_filter. Input	scene.uff	Pending
lid_filter. Input	orbit.xml	Pending
lid_filter. Configuration	lid_filter_config.xml	$ECSIM_HOME/sessions/Lidar/0-Lidar/lid_filter_config.xml
lidar. Input	lid_filter_out.nc	Pending

© DEIMOS Space S.L., 2007

Model and type	File name	File instance
lidar. Input	orbit.xml	Pending
lidar. Configuration	lidar_config.xml	$ECSIM_HOME/sessions/Lidar/0-Lidar/lidar_config.xml
lid_l2a. Input	lidar_out.nc	Pending
lid_l2a. Configuration	lid_l2a_config.xml	$ECSIM_HOME/sessions/Lidar/0-Lidar/lid_l2a_config.xml

5.2.3. Steps

- Select from the repository tab of the side bar, a session named "Lidar". Right-click on it and select the "Edit" option.
- Browse every field of the input, configuration, parameters, output and tools tabs to edit the values you desire. You can use the files in the table above to perform a sample execution.
- Once everything is as your wishes, click on the "Run" button to start the execution.
- The execution will start and you can see its progress status.
- Once the simulation ends, all the post-processing tools will start and you can see how the scene will be obtained.

5.3. Synergistic Instrument Retrieval

5.3.1. Operation objectives

The objective is to perform a full end-to-end simulation for the EarthCARE, simulating a real situation, the behaviour of several on-flight instruments and the retrieval processing, and then comparing the output obtained at ground segment with the original scene.

In this case the simulation will use both the ATLID and CPR instruments and will retrieve a combination of both outputs.

5.3.2. Pre-conditions

A previously defined session called "Synergy" exists in the repository. This session includes:

- One simulation, named "Synergy". This simulation starts from the scene creator to the retrieval stage and it contains all the models related with only the ATLID and CPR instruments.
- A list of product tools to visualize the results. These tools must be well defined and accessible for the system.

The following set of input, auxiliary and configuration files will be defined:

Model and type	File name	File instance
scene_creator_xml. Input	scene_inp.xml	$ECSIM_HOME/sessions/Synergy/0-Synergy/scene_inp.xml
orbit_dms. Input	scene.uff	Pending

© DEIMOS Space S.L., 2007

Model and type	File name	File instance
orbit_dms. Configuration	orbit_dms_config.xml	$ECSIM_HOME/sessions/Synergy/0-Synergy/orbit_dms_config.xml
rad_filter. Input	uff_adjust.xml	$ECSIM_HOME/sessions/Synergy/0-Synergy/uff_adjust.xml
rad_filter. Input	scene.uff	Pending
rad_filter. Input	orbit.xml	Pending
rad_filter. Configuration	rad_filter_config.xml	$ECSIM_HOME/sessions/Synergy/0-Synergy/rad_filter_config.xml
lid_filter. Input	uff_adjust.xml	$ECSIM_HOME/sessions/Synergy/0-Synergy/uff_adjust.xml
lid_filter. Input	scene.uff	Pending
lid_filter. Input	orbit.xml	Pending
lid_filter. Configuration	lid_filter_config.xml	$ECSIM_HOME/sessions/Synergy/0-Synergy/lid_filter_config.xml
radar. Input	rad_filter_out.nc	Pending
radar. Input	orbit.xml	Pending
radar. Configuration	radar_config.xml	$ECSIM_HOME/sessions/Synergy/0-Synergy/radar_config.xml
lidar. Input	lid_filter_out.nc	Pending
lidar. Input	orbit.xml	Pending
lidar. Configuration	lidar_config.xml	$ECSIM_HOME/sessions/Synergy/0-Synergy/lidar_config.xml
rad_l2a. Input	radar_out.nc	Pending
rad_l2a. Configuration	rad_l2a_config.xml	$ECSIM_HOME/sessions/Synergy/0-Synergy/rad_l2a_config.xml
lid_l2a. Input	lidar_out.nc	Pending
lid_l2a. Configuration	lid_l2a_config.xml	$ECSIM_HOME/sessions/Synergy/0-Synergy/lid_l2a_config.xml
lid_rad_l2b. Input	lid_l2a_out.nc	Pending
lid_rad_l2b. Input	rad_l2a_out.nc	Pending
lid_rad_l2b. Configuration	lid_rad_l2b_config.xml	$ECSIM_HOME/sessions/Synergy/0-Synergy/lid_rad_l2b_config.xml

5.3.3. Steps

- Select from the repository tab of the side bar, a session named "Synergistic". Right-click on it and select the "Edit" option.

© DEIMOS Space S.L., 2007

- Browse every field of the input, configuration, parameters, output and tools tabs to edit the values you desire. You can use the files in the table above to perform a sample full end-to-end simulation.
- Once everything is as your wishes, click on the "Run" button to start the execution.
- The execution will start and you can see its progress status.
- Once the simulation ends, all the post-processing tools will start and you can see how the scene will be obtained in the case of operating with only the CPR and ATLID instruments.
- Other operations you can do upon the execution's end are to check every other generated file or the final execution report where you can find detailed information about it.

5.4. Experiment with Multiple runs

5.4.1. Operation objectives

The objective is to perform an iterative simulation of a certain simulation (in this case, a RADAR simulation), testing the changes on a certain parameter.

5.4.2. Pre-conditions

A previously defined session called "Radar" exists in the repository. This session includes:

- One simulation, named "Radar". This simulation starts from the scene creator to the instruments stage and it contains all the models related with the CPR instrument.
- A list of product tools to perform the graphical comparison and to visualize the results. These tools must be well defined and accessible for the system.

The following set of input, auxiliary and configuration files will be defined:

Model and type	File name	File instance
scene_creator_xml. Input	scene_inp.xml	$ECSIM_HOME/sessions/Radar/0-Radar/scene_inp.xml
orbit_dms. Input	scene.uff	Pending
orbit_dms. Configuration	orbit_dms_config.xml	$ECSIM_HOME/sessions/Radar/0-Radar/orbit_dms_config.xml
rad_filter. Input	uff_adjust.xml	$ECSIM_HOME/products/uff_adjust.xml
rad_filter. Input	scene.uff	Pending
rad_filter. Input	orbit.xml	Pending
rad_filter. Configuration	rad_filter_config.xml	$ECSIM_HOME/sessions/Radar/0-Radar/rad_filter_config.xml
radar. Input	rad_filter_out.nc	Pending
radar. Input	orbit.xml	Pending
radar. Configuration	radar_config.xml	$ECSIM_HOME/sessions/Radar/0-Radar/radar_config.xml

© DEIMOS Space S.L., 2007

5.4.3. Steps

- Select the "HorSampling" parameter from the radar model and access to the "Iterate parameters" function.
- Edit the parameter's values and input "200.0 300.0 400.0". Accept the changes and run the simulation.
- You can see that the radar instrument is executed three times (one per each configuration) so you can compare data from the three different situations.

5.5. E2E simulation

5.5.1. Operation objectives

The objective is to perform an end-to-end simulation for the EarthCARE, simulating a real situation and the retrieval processing, and then comparing the output obtained at ground segment with the original scene.

© DEIMOS Space S.L., 2007

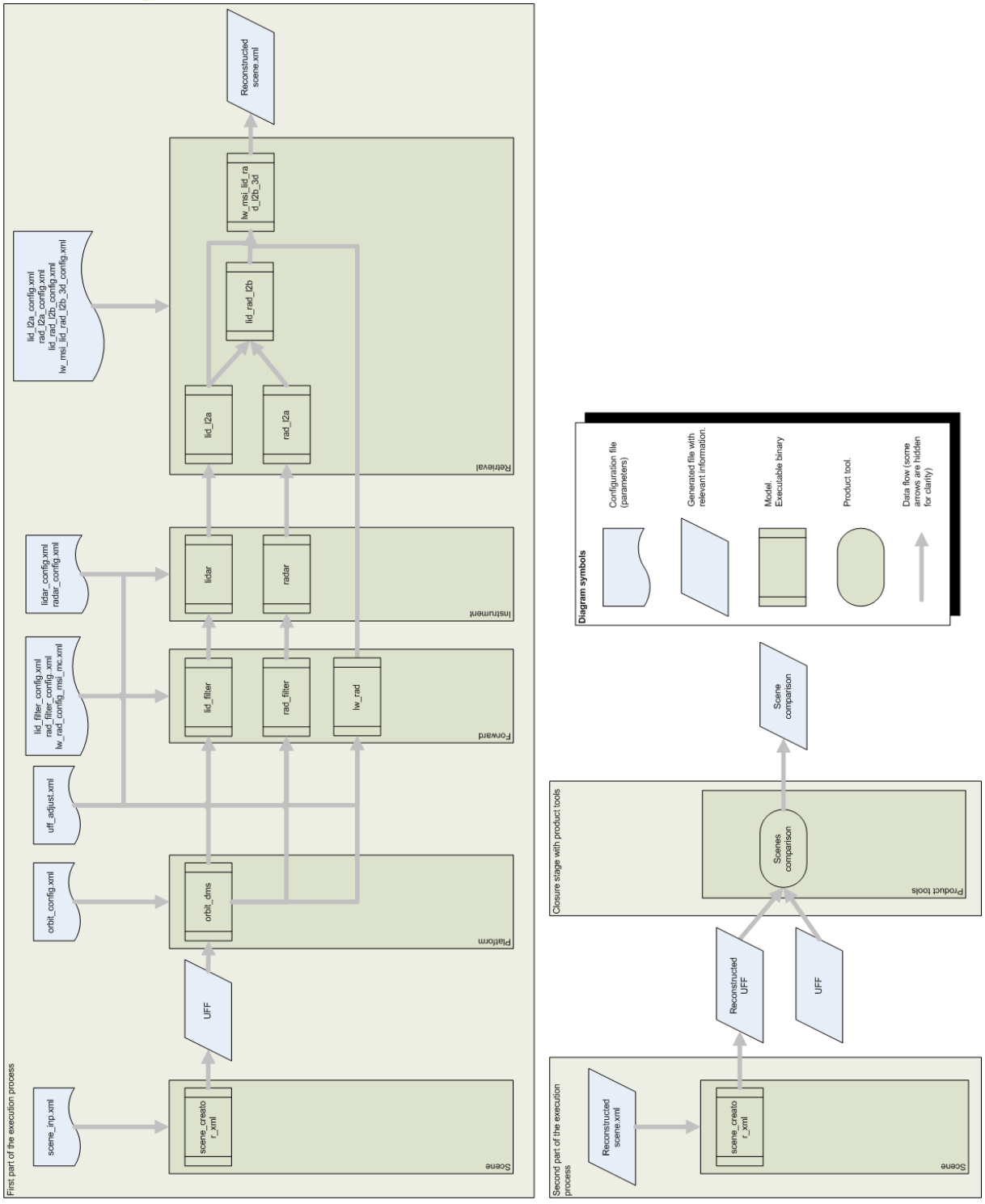


Figure 5-1: End-to-end simulation schema

© DEIMOS Space S.L., 2007

5.5.2. Pre-conditions

A previously defined session called "E2E" exists in the repository. This session includes:

- A set of two already-defined simulations, named "Complete" and "SceneCreation". The former will create a XML description of a reconstructed scene from a forward and retrieval simulation and the later will create an UFF file out of the reconstructed scene.

The following set of input, auxiliary and configuration files will be defined:

Model and type	File name	File instance
scene_creator_xml. Input	scene_inp.xml	$ECSIM_HOME/sessions/E2E/0-Complete/scene_inp.xml
orbit_dms. Input	scene.uff	Pending
orbit_dms. Configuration	orbit_dms_config.xml	$ECSIM_HOME/sessions/E2E/0-Complete/orbit_dms_config.xml
lw_rad. Input	uff_adjust.xml	$ECSIM_HOME/sessions/E2E/0-Complete/uff_adjust.xml
lw_rad. Input	scene.uff	Pending
lw_rad. Input	orbit.xml	Pending
lw_rad. Configuration	lw_rad_config.xml	$ECSIM_HOME/sessions/E2E/0-Complete/lw_rad_config.xml
rad_filter. Input	uff_adjust.xml	$ECSIM_HOME/sessions/E2E/0-Complete/uff_adjust.xml
rad_filter. Input	scene.uff	Pending
rad_filter. Input	orbit.xml	Pending
rad_filter. Configuration	rad_filter_config.xml	$ECSIM_HOME/sessions/E2E/0-Complete/rad_filter_config.xml
lid_filter. Input	uff_adjust.xml	$ECSIM_HOME/sessions/E2E/0-Complete/uff_adjust.xml
lid_filter. Input	scene.uff	Pending
lid_filter. Input	orbit.xml	Pending
lid_filter. Configuration	lid_filter_config.xml	$ECSIM_HOME/sessions/E2E/0-Complete/lid_filter_config.xml
radar. Input	rad_filter_out.nc	Pending
radar. Input	orbit.xml	Pending
radar. Configuration	radar_config.xml	$ECSIM_HOME/sessions/E2E/0-Complete/radar_config.xml
lidar. Input	lid_filter_out.nc	Pending
lidar. Input	orbit.xml	Pending

© DEIMOS Space S.L., 2007

Model and type	File name	File instance
lidar. Configuration	lidar_config.xml	$ECSIM_HOME/sessions/E2E/0-Complete/lidar_config.xml
rad_l2a. Input	radar_out.nc	Pending
rad_l2a. Configuration	rad_l2a_config.xml	$ECSIM_HOME/sessions/E2E/0-Complete/rad_l2a_config.xml
lid_l2a. Input	lidar_out.nc	Pending
lid_l2a. Configuration	lid_l2a_config.xml	$ECSIM_HOME/sessions/E2E/0-Complete/lid_l2a_config.xml
lid_rad_l2b. Input	lid_l2a_out.nc	Pending
lid_rad_l2b. Input	rad_l2a_out.nc	Pending
lid_rad_l2b. Configuration	lid_rad_l2b_config.xml	$ECSIM_HOME/sessions/E2E/0-Complete/lid_rad_l2b_config.xml
lw_msi_lid_rad_l2b_3d. Input	msi_l2a_out.nc	Pending
lw_msi_lid_rad_l2b_3d. Input	lid_rad_l2b_out.nc	Pending
lw_msi_lid_rad_l2b_3d. Configuration	lw_msi_lid_rad_l2b_3d_config.xml	$ECSIM_HOME/sessions/E2E/0-Complete/ lw_msi_lid_rad_l2b_3d_config.xml
scene_creator_xml. Input	scene_inp.xml	/../0-Complete/recon_scene_config.xml[10]

5.5.3. Steps

- Select from the repository tab of the side bar, a session named "E2E". Right-click on it and select the "Edit" option.
- Browse every field of the input, configuration, parameters, output and tools tabs to edit the values you desire. You can use the files in the table above to perform a sample full end-to-end simulation.
- Once everything is OK, click on the "Run" button to start the execution.
- The execution will start and you can see its progress status.
- Once the simulation ends, you can access to the scene comparison feature in the main menu and see a graphical comparison between the original scene and the reconstructed scene.
- Other operations you can do upon the execution's end are to check every other generated file or the final execution report where you can find detailed information about it.

[10] This is the way to use the output of a simulation as input for another.

© DEIMOS Space S.L., 2007

5.6. List of available scenes

ECSIM distribution includes some scene definitions for testing purposes in the final delivery. Here is their description:

- "standard_test_scene". Standard scene for testing purposes. Defines two scattering regions: one with a background sulfate areosol layer and another which contains a gamma-type single mode water distribution layer.
- "scene_fire_fractal". Scene included in the "worst-case scenario". Includes a fractal ice cloud between 7 and 10 km present from x=0-20 and y=0-20 km.

5.7. List of available tools

ECSIM distribution includes some already-defined product tools. Here is their description:

Product tool	Description	Extension	Executable
extract_env	This program reads a 3-D domain from a UFF file and extracts various averaged quantities and outputs them to the specified output file.	uff	tools/product_tools/bin/extract_env
extract_quantity	Extract quantity is used to extract information from a UFF file along a straight line between (x1, y1) and (x2, y2). The program will read the UFF file as well as any relevant information from the scattering libraries in order to build-up the requested data.	uff	tools/product_tools/bin/extract_quantity
extract_quantity_3d	This program reads a 3-D domain from a UFF file and extracts various averaged quantities and outputs them to the specified output file.	uff	tools/product_tools/bin/extract_quantity_3d
extract_quantity_hor	This program reads an entire UFF file and extracts various column integrated quantities and outputs them to the specified output file.	uff	tools/product_tools/bin/extract_quantity_hor
gthumbviewer	Image viewer	gif	gthumb

© DEIMOS Space S.L., 2007

Product tool	Description	Extension	Executable
l1_l2_compare	L1_L2_compare is used to compare the quantities in two different 2-D ncdf files. The program will read the data from the netcdf files and return the differences (absolute and relative of file1-file2) in the output netcdf file… It reads in data from the L1_L2_rebin tool. This means that there will be a predefined slab in the data stream of the input file and this argument is not needed here.	nc	tools/product_tools/ bin/l1_l2_compare
l1_l2_rebin	L1_L2_rebin is used to change the resolution of the lidar and radar output files. The program will read the data from the netcdf file and return the data in the new resolution in the output netcdf file. It will retrieve the along_track and height for vertical slabs (made with extract_quantity) and the x-scene and y-scene for the horizontal slabs (made with extract_quantity_hor).	nc	tools/product_tools/ bin/l1_l2_rebin
NetCDF differencer	ncdiff subtracts variables in file_2 from the corresponding variables (those with the same name) in file_1 and stores the results in file_3	nc	ncdiff
ncBrowse	NetCDF viewer	nc	ncBrowse (must be in the system PATH)
plot_3d	This program reads a 3-D file from extract_quantity_3d and produces 3-D iso-surface plot along with a 2-D colour scale plot slice.	nc	tools/pgplot_tools/ bin/plot_3d
plot_hor	This program reads a 2-d file from extract_quantity_hor and produces 2-d image plot.	nc	tools/pgplot_tools/ bin/plot_hor

© DEIMOS Space S.L., 2007

Product tool	Description	Extension	Executable
plot_profile	This program reads a 2-D (height-vs-along_track) data file generated by extract_quantity, lidar, radar etc… and produces a profile plot corresponding to a specific along_track point. Multiple data files and quantities are possible and can be plotted using different line types, thicknesses, and colours. A legend box is also plotted.	nc	tools/pgplot_tools/bin/plot_profile
plot_slice	This program reads a 2-D (height-distance) data file generated by extract_quantity, lidar, radar etc… and produces a 2-D image plot.	nc	tools/pgplot_tools/bin/plot_slice
uff_averager	This program reads an UFF file and using simple averaging it outputs a UFF file of lesser resolution. Size distributions are averaged on a bin-by-bin basis over the desired domain.	uff	tools/product_tools/bin/uff_averager
uff_merger	This program merges two UFF files in the vertical coordinate.	uff	tools/product_tools/bin/uff_merger
View PS files	PS files viewer	ps	evince
View sh	SH files viewer	sh	gedit
View XML	XML files viewer	xml	gedit
XML differencer	Finds the differences between two xml files	xml	diff

© DEIMOS Space S.L., 2007

End of document

© DEIMOS Space S.L., 2007